POLITECNICO DI TORINO

Master Degree course in Mechatronic Engineering

Master Degree Thesis

Energy-Efficient Adaptive Cruise Control: An Economic MPC Framework Based on Constant Time Gap

Supervisors
Prof. Michele Pagone
Dott. Lorenzo Calogero
Prof. Carlo Novara
Prof. Alessandro Rizzo

Candidate
Paolo Ammaturo

Academic Year 2024-2025

Abstract

In recent years, the rising concerns about climate change have increasingly driven automotive companies to invest in electric vehicle (EV) development. While EVs stand as a promising solution to enable a more sustainable form of transportation, they still present inherent limitations compared to traditional ones, particularly in terms of driving range. To address these challenges, this thesis develops a novel nonlinear Economic Model Predictive Control (EMPC) strategy based on a Constant Time Gap (CTG) approach for adaptive cruise control, aiming to simultaneously achieve optimal control performance and energy efficiency.

The main challenge in designing the EMPC control strategy lies in managing multiple concurrent control objectives. This thesis approaches such a challenge by employing a carefully structured cost function for the optimal control problem, encompassing various aspects among which the temporal evolution of the battery state of charge, the ego vehicle energy consumption, and the ahead vehicle behavior. These objectives typically concur with each other: aggressive following behavior might lead to higher energy consumption and faster battery depletion, while strict energy saving could compromise the cruise control performance.

Using experimental data collected from a real vehicle (a “Fiat 500e”), different cost functions are designed and compared to evaluate their effectiveness in carrying out the cruse control task while preserving energy efficiency. The main goal is to find an optimal balance between these two crucial metrics.

The cost functions are tested in increasingly complex scenarios. First, validation is carried out considering constant and sinusoidal velocity profiles. Then, standard driving cycles – based on real-world vehicle operation – are employed, among which the WLTP class 3 cycle.

Simulation results show that the CTG-based EMPC strategy achieves better energy efficiency that traditional MPC approaches while proficiently attaining the cruise control task. The research work carried out in this thesis demonstrates how CTG principles can be successfully integrated within economic objectives for EVs control. While the immediate focus of this study are EVs, the general formulation of the EMPC framework makes it adaptable to several other power management scenarios. The cost function structure, which balances energy consumption and control performance, could find potential application in other domains, like different hybrid powertrain configurations (e.g., fuel cell hybrid electric vehicles), smart grids, and HVAC systems. Each of these scenarios presents similar trade-offs between resource utilization and system performance, suggesting that the proposed EMPC approach could have a wider application beyond EVs.

Contents

List of Tables

List of Figures

Acronyms and Symbols

Acronyms

ACC	Adaptive Cruise Control
BEV	Battery Electric Vehicle
BMS	Battery Management System
CTG	Constant Time Gap
CoG	Center of Gravity
EMPC	Economic Model Predictive Control
EV	Electric Vehicle
HVAC	Heating, Ventilation, and Air Conditioning
LTV-MPC	Linear Time-Varying Model Predictive Control
Maas	Mobilty as a Service
MCU	Motor Control Unit
MIMO	Multiple Input Multiple Output
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
OBC	On-Board Charger
PD	(Proportional-Derivative) [controller]
PMSM	Permanent Magnet Synchronous Motor
SOC	State of Charge
SOH	State of Health
VCU	Vehicle Control Unit
WLTC	Worldwide harmonized Light vehicles Test Cycle
WLTP	Worldwide harmonized Light vehicles Test Procedure

Symbols

ω	Angular velocity
τ	Torque
τ_{gb}	Gearbox ratio
ζ	State of Charge (SOC)
$\dot{\zeta}$	Rate of change of SOC
η	Efficiency (generic)
η_{em}	Electric motor efficiency
η_{gb}	Gearbox efficiency
η_{inv}	Inverter efficiency
η_{b}	Battery power converters
η_{c}	Coulomb efficiency
α	Weight parameter (velocity)
β	Weight parameter (power)
γ	Weight parameter (terminal)
ρ	Air density
F	Force (generic)
F_{roll}	Rolling resistance force
F_{vis}	Aerodynamic resistance
F_{grade}	Grade force
F_{w}	Wheel force
ℓ	Stage cost
J	Cost function
P	Terminal cost weight matrix
P_{em}	Electric motor power
P_{b}	Battery power
P_{ref}	Reference power
Q	State error weight matrix
Q_{nom}	Nominal battery capacity
R	Control input weight matrix
R_{o}	Internal resistance
R_{b}	Battery internal resistance
R_{c}	Cell internal resistance
M	Weighting matrix
M_{v}	Vehicle mass
T_{em}	Electric motor torque
T_{w}	Wheel torque
V_{oc}	Open circuit voltage
V_{B}	Battery voltage
a	Acceleration
a_{v}	Vehicle acceleration
f_0	Static rolling coefficient
g	Gravitational acceleration
h	Time gap
k	Controller gain
r_{w}	Wheel radius
u	Control input
v	Velocity
v_{ref}	Reference velocity
x	State vector

Chapter 1

Introduction

The way we think about cars has change a lot. Before, speed was everything, we want to move from point A to point B as quickly as possible. Now, it's different. We don't care only about the speed but also taking into account the environmental impact. This shift happened because climate change effects are becoming visible in our daily lives. Floods, higher than average temperatures and melting glaciers are the order of the day.

For this reason, climate change has emerged as a defining challenge for our era, with the transportation sector contributing significantly to global emissions [18]. Reduce the use of fossil fuels has therefore become essential. For this reason, politicians are taking action, such as the European Union, with regulation 2023/851 [5]. This regulation sets clear targets. The goal is zero emissions by 2035, through a progressive reduction of emissions.

Research Context and Problem Statement

My thesis addresses a critical challenge in electric vehicle technology: energy optimization. While EVs offer a cleaner alternative to traditional combustion engines, they still face practical limitations. Early models struggled with three main problems: they could not drive far, took too long to charge and cost too much. To help solve these problems, policy makers are actively supporting the transition to sustainable mobility and in particular to electric mobility, through various incentives. These include support for the use of public transport like Mobility as a Service (MaaS), and incentives for electric vehicles like direct purchase incentives, exemption from property tax, free parking in paid areas and restrictions on high-emission vehicle. Modern EVs show significant improvements. Battery technology has advanced, allowing for greater energy storage and the power management have become more sophisticated, thanks to smarter control systems ensure that power is used efficiently. Charging stations have become common and driving ranges have been extended. Despite these advances, the quest for efficiency continues to drive innovation. My thesis address this challenge and i focus on improve how electric vehicle (EVs) use their power during Adaptive Cruise Control (ACC) scenario. I will use Economic Model Predictive Control (EMPC) in order to try to maximize the efficiency of power usage for extending vehicle range, improving overall performance, while following in best manner the vehicle in front.

Research Contribution

My main contribution is the development of a nonlinear economic model predictive control (EMPC) for the EV powertrain system using CasADi [1]. This controller aims to:

- Optimize battery consumption patterns

- Follow the vehicle in front as best as possible, avoiding sudden accelerations

Methodology

My methodology focuses on EV control optimization. Instead of using complex models, I focused on approach that prioritizes control system development. My methodological approach emerged directly from my background in mechatronics/computer engineering, which give me a distinct perspective on approaching EV control optimization challenges. While vehicle dynamics present intricate complexity, my academic background led me to prioritize control system innovation over the vehicle modeling. For this reason, i decide to employ in my work a two-axle vehicle model, a choice that takes into account my academic background and my research priorities. The two-axle model, which is a simple version of multi- body dynamics representation often used in automotive engineering, provides me the ideal basis for this problem. This model captures the essential longitudinal dynamics needed to develop the control system without considering the complexity of full vehicle dynamics, like suspension dynamics, the road-tire interface phenomena, inertial effects, etc. Using data from Politecnico di Torino's Fiat 500e, this provide me a real-world parameters, grounding the theoretical work in practical application. The methodology developed along three main threads:

- The vehicle modeling phase focused on extracting the most relevant dynamics for control purposes deliberately abstracting away complexities that wouldn't substantially impact controller performance.
- System analysis examined the interplay between mechanical and electrical subsystems, with particular attention to aspects most relevant to control design.
- The simulation phase, where I implemented and tested the control strategy using MATLAB [12] and Simulink. I followed a progressive approach: first, I tested the controller with a constant speed reference to verify basic functionality. Once that worked correctly, I moved to a sinusoidal velocity reference.

In the end I tested the controller with the WLTP3 cycle, which represents realistic driving condition, that presents various acceleration phase, different steady-state speed and multiple deceleration.

Thesis Organization

The thesis follows a logical flow starting with basic concepts. First, I explain how EVs work and their structure, then I develop a model. After that, I show the implementation of my controller and discuss what I found. I organize my chapters like this:

- **Chapter II - Fundamentals of Electric Vehicles**: This chapter introduces the most important component of EV, provides a general understanding of electric vehicle and how these elements works together to give readers a foundation of how electric vehicles works.

- **Chapter III - System Model**: In this section, I develop the mathematical framework for vehicle dynamics and battery behavior. I focus on creating a practical model that balances accuracy with computational efficiency, using data from the Fiat 500e as reference.

- **Chapter IV - Model Predictive Control: From Traditional to Economic Frameworks:** Here, I explore MPC theory and its evolution toward economic objectives. I explain the shift from traditional tracking formulations to economic approaches that optimize operational costs.

- **Chapter V - Economic MPC Implementation**: The core of my contribution. Here, I detail the design process, cost function development, constraint formulation and practical implementation challenges encountered during controller development.

- **Chapter VI - Simulation Results and Conclusion**:The final chapter presents simulation results across different driving scenarios. I analyze performance differences between control strategies, discuss parameter selection trade-offs, and draw conclusions about the effectiveness of Economic MPC for electric vehicles.

Literature Review

Recent years have seen significant advantages in EV technology and control systems. The research landscape spans different areas, from powertrain optimization to energy management strategies.

A key focus has been the development of model predictive control (MPC) strategies. For example, Borhan et al. explored MPC for power management in hybrid vehicles, demonstrating how MPC can effectively manage power split between different energy sources [3]. In this work, they proposed two distinct MPC approaches. A first strategy that consider a linear time-varying MPC (LTV-MPC), which proved comparable to existing control methods. But this approach present a limitations of linearization in capturing complex powertrain dynamics, so taking into account this consideration, they develop a nonlinear MPC (NMPC). This approach that show an improvement in fuel economy. However, their NMPC approach required approximately double the computational time of LTV-MPC. Building on this foundation, Pereira et al. demonstrated the potential of nonlinear MPC for real-time applications [14]. This research address practical implementation challenges, particularly in computational efficiency, because they run the NMPC on low-cost hardware, like the hardware that is inside the vehicles. Their work provided a valuable insight into making complex control strategies feasible for real-world application.

Another crucial research direction is the energy management. Different teams of researcher have tried different approach. He et al. developed a fuzzy-logic based system for energy management [11]. While, for example Zhang et al. in their research explored the effectiveness of equivalent consumption minimization strategy in a hybrid tram system [22].

Adaptive control is also another exciting area of research. Here, Xin and their propose an innovative approach. In their research, they proposed an MPC approach that includes online mass estimator [21]. The idea on the basis is the following, as passengers get in and out or cargo is loaded, the vehicle's mass changes. This changes affect how much energy is used by the vehicle. By detecting the real-time change in mass, the controller can modify its strategy. Following this idea, they are able to achieve better energy efficiency compared to traditional controllers that assume a fixed mass.

The evolution of alternative power sources has added another dimension to EV research. Of particular interest is fuel cell technology. In this area, Qiu et al. conduced an analysis of multi-stack fuel cell systems [15]. Their work revealed crucial insights about system architecture and highlighted how optimization at the system level significantly impacts overall efficiency.

My research builds on these foundation. I focus on developing an Economic MPC strategy [10] that handles both energy optimization and ACC behavior [20]. For simulation and validation, I use the World Harmonized Light-duty Test Cycle (WLTC) [17], specifically the WLTP class 3 cycle, which is the appropriate class for my vehicle, the Fiat 500e. This cycle offers a comprehensive speed profile that spans different driving conditions from urban to highway cruising. This type of variousness into cycle, makes it particularly well-suited for evaluating controller performance across realistic driving scenarios. Its structure thoughtfully combines gentle acceleration phases with more demanding high-speed segments followed by deceleration phase. This varied profile creates an ideal testbench for the proposed control strategy and energy optimization scheme. During high-speed segments, the cycle demands significant power from the motor, while the deceleration phases enable regenerative braking to replenish battery charge. This natural alternation between power consumption and regeneration allows for thorough testing of the energy management capabilities of the control system.

Chapter 2

Fundamentals of electric vehicles

In this chapter, I present the main components of an electric vehicle powertrain interact with each other. My focus is on understanding how these elements works together and give to the reader a basis of EVs.

The shift towards sustainable transportation is transforming the automotive industry, much more vehicles now use alternative power sources. This affects not only private transport but also public transportation, such as buses.

The demand for sustainable transportation has grown rapidly. This growth has led to more vehicles with lower emission [19]. These vehicles use alternative fuel types, such as hybrid and electric powertrain. This shift affects both private and public transportation and, the growing market for electric and hybrid vehicles has prompted automotive companies to direct significant investments toward electric powertrain development.

2.1 Battery System

Battery Management Systems (BMS) constantly monitors various parameters of the battery cells. It tracks the State of Charge (SOC) and the State of Health (SOH). This task of monitoring the general health of the battery is fundamental. This control allows the system to always operate in the optimal condition. Temperature represents a crucial element for optimal battery functioning, as extreme values (both too high and low) can significantly affect the battery life. Moreover, excessive battery temperatures pose significant safety risks, potentially leading to thermal runaway events. In addition, the BMS performs another crucial task. Lithium-ion batteries are made up of several cells connected in series or in parallel with each other. Each cell charges or discharges at different speeds. This variation can cause cell overcharging, leading to degradation, temperature increases, and potential swelling. The BMS maintains balanced cell charges to avoid these issues.

2.2 Power conversion

The EV powertrain contains two key power conversion devices: an inverter and a converter. The inverter transforms DC power from the battery pack into AC power needed by the electric motor. During this conversion, some part of energy is lost as heat.

When the vehicle brakes, the system can recover energy through regenerative braking . In this case, the motor acts as generator. It produces AC power, which the inverter converts back to DC to recharge the battery pack. This energy recovery, improves the overall efficiency.

The power converter efficiency shows up in the battery current equation (eq. (3.17)) through the parameter η_b. In my model, I keep η_b constant to simplify things. But even small changes in efficiency can affect the range of the vehicle.

Another task performed by the power conversion systems is to step down the

DC voltage of the battery pack, which usually ranges from 100 to 400 V to much lower values 12-24 V, this is performed by a DC/DC converter. This lower voltage is used by the services inside the car, such as air conditioning, radio, screens, etc.

I kept this setup simple in my study. But in real cars, managing these voltage conversion efficiently is crucial for the whole system.

2.3 Motor Control Unit and Vehicle Control Unit

The Vehicle Control Unit (VCU) works in synergy with the Motor Control Unit (MCU) in the electric vehicle. The VCU acts as the brain of the car. It manages the overall operations and monitors the status of the vehicle systems. The VCU has three main tasks. First, it handles powertrain management by sending power request to the MCU. Second, it controls the vehicle dynamics, stability, through real-time sensors. This includes managing traction control and brake force distribution. Third, it runs diagnostic operations to ensure that vehicle works in safety conditions. It does this by using fault detection algorithms to monitor all subsystems.

On the other hand, the MCU focuses on motor operation. It receives power request from VCU and convert them into specific torque commands. This process determines how much battery power should go to the electric motor. This is done through the control of the inverter.

These two units form a critical control hierarchy. The VCU makes high-level decisions about vehicle operation. The MCU then implements these decision through precise motor control.

2.4 Thermal Management

Temperature control is a fundamental feature for electric vehicles, with particular emphasis on two key areas: the battery pack and the electric motor. The Fiat 500e model that I'm considering in this thesis, uses a liquid cooling system [16].

The coolant circulates through the battery pack, dissipating heat generated during operation. According to the vehicle's user manual [16],the battery's operating range is between $-30°C$ to $50°C$. Temperatures that are too high or too low could ruin the cell. The cooling system maintains optimal operating conditions. This affects both performance and battery life.

Beyond battery management, the thermal system also regulates the temperature of the electric motor. Heat generation becomes particularly pronounced during high-power demands. Without adequate cooling, motor efficiency would deteriorate, affecting overall vehicle performance.

The power electronics, specifically the inverter and converter, also require thermal management. Although their cooling demands are less intensive than the battery pack's requirements, these components generate heat during operation that must be dissipated.

Temperature sensors are placed inside these components and provide real-time data for thermal management. This monitoring becomes crucial during rapid charging phases, where heat generation intensifies significantly. During these periods, the cooling system operates at higher capacity to prevent thermal stress on the battery cells.

This comprehensive approach to thermal management directly impacts vehicle efficiency and component longevity. By maintaining optimal operating temperatures across all systems, the thermal management system plays a vital role in ensuring reliable vehicle performance.

2.5 Electric Motor and Transmission

The electric motor acts as the core component in EVs. It transforms electrical energy from the battery into mechanical power. Tn my thesis, I focus on the Fiat

500e's Permanent Magnet Synchronous Motor (PMSM). This type of motor is very common in electric cars, because it is very efficient and provide good torque even at low speeds. Which is very useful in urban scenarios.

The transmission in EVs differs significantly from traditional combustion engines. Electric motors can deliver maximum torque at zero speed [7]. My study of the Fiat 500e reveals a fixed gear ratio of 9.6:1 with 97% gearbox efficiency (Table 3.1). These parameters determine how motor power transfers to the wheels.

Motor efficiency varies with operating conditions. Figure 3.1 demonstrates how speed and torque affect efficiency levels. This relationship significantly influences energy consumption. In my controller design, I implemented a practical simplification: I treat efficiency as constant during the prediction horizon. This approach uses current efficiency values to determine subsequent torque commands, reducing computational complexity while maintaining reliability.

During normal driving, it provides propulsion. During braking, it becomes a generator. This generative braking process recovers kinetic energy, converting it back to electrical form for battery charging. However, that energy recovery remains partial due to system losses, primarily through heat dissipation.

These insights into motor and transmission behavior form a crucial foundation for my control system design work in later chapters.

2.6 On-Board Charger

The On-Board Charger (OBC) is a fundamental component in EVs. It converts AC power from the electrical grid to DC power that the battery can use. In the Fiat 500e, like in the modern EVs, the OBC manages both standard home charging and faster charging stations. The OBC work closely with Battery Management System. Together, they ensure safe and efficient charging. While the OBC is crucial for EV operation, I don't focus on it in my control system design. My work centers on energy management during vehicle operation rather than charging processes.

The interactions between these components form a complex but organized system. Figure 2.1 shows how these elements work together. This diagram helps us understand the flow of power, control signals, and thermal management in an EV.

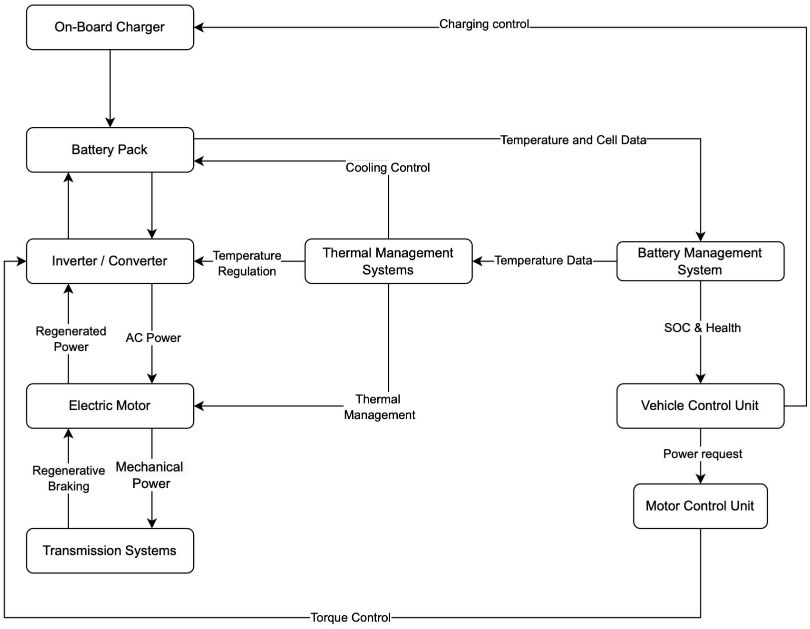

Figure 2.1: Block diagram illustrating the key components and interactions within an electric vehicle powertrain system. The diagram shows the flow of power, control signals, and thermal management between major subsystems

Chapter 3

System model

Electric vehicles require sophisticated control strategies, which in turn need accurate models for their development. This chapter introduces the fundamental concepts I use throughout my work, aiming to build a clear understanding of the interactions between mechanical and electrical systems in the vehicle.

The organization of this chapter reflects the natural progression of my analytical framework. I begin with the mechanical aspects of the test vehicle, a Fiat 500e owned by Politecnico di Torino. After analyzing several modeling approaches, I chose to use a simplified dynamics model - the "two-axle vehicle body representation". This choice stems from a practical consideration: since my thesis focuses on controller development rather than advanced vehicle dynamics,I needed a model that balances realistic behavior with manageable complexity.

This simplified approach proved particularly useful because:

- It captures essential vehicle dynamics without excessive computational overhead.

- It provides sufficient accuracy for control design purposes.

- It allows me to focus more resources on the control aspects of my work

Following my analysis of the vehicle's longitudinal dynamics, I turned my attention to the electrical system. Understanding the complex relationship between battery charge-discharge cycles and vehicle performance.

3.1 Vehicle Parameters

The vehicle parameters used in this thesis are derived from the Fiat 500e model, as provided by Professor Pagone, and are shown in Table 3.1 .

Parameter	Symbol	Value	Unit
Vehicle Mass	M_{v}	1400	kg
Wheel radius	r_{w}	0.3	m
Frontal area	A_{f}	2.15	m^2
Static rolling coefficient	f_0	4.5	N/kN
Drag coefficient	C_{d}	0.33	-
Gear ratio	τ_{gb}	9.6	-
Gearbox efficiency	η_{gb}	0.97	-

Table 3.1: Vehicle parameters of the Fiat 500e

3.2 Longitudinal Dynamics Model

The longitudinal vehicle dynamics can be described using a backward approach. The model is based on a two-axle vehicle body representation, where all forces are assumed to be applied at the center of gravity (CoG).This approach allows me to simplify the dynamic analysis while maintaining model accuracy for longitudinal motion studies. Starting from a desired reference acceleration,I calculate the required wheel torque by considering all forces acting on the vehicle during motion.

The main forces affecting longitudinal dynamics are: inertial force ($M_{\mathrm{v}}a_{\mathrm{ref}}$), grade force ($F_{\mathrm{grade}}$) due to road inclination, aerodynamic resistance (F_{vis}) which has a quadratic relationship with velocity and rolling resistance (F_{roll}) mainly dependent on vehicle weight. The required wheel torque T_{w} is calculated by multiplying the sum of these forces by the radius of the wheel:

$$T_{\mathrm{w}} = (M_{\mathrm{v}} a_{\mathrm{ref}} + F_{\mathrm{grade}} + F_{\mathrm{roll}} + F_{\mathrm{vis}}) r_{\mathrm{w}} \tag{3.1}$$

where:

- The grade force is given by:

$$F_{\mathrm{grade}} = M_{\mathrm{v}}\, g\, \sin\alpha \tag{3.2}$$

- The rolling resistance is expressed as:

$$F_{\mathrm{roll}} = M_{\mathrm{v}}\, g\, f_0\, \mathrm{sgn}\ v \tag{3.3}$$

- The aerodynamic resistance force is:

$$F_{\mathrm{vis}} = \frac{1}{2}\, \rho\, A_{\mathrm{f}}\, C_{\mathrm{d}}\, v^2 \tag{3.4}$$

  where $\rho = 1.25\ kg/m^3$ is the air density

In my analysis, I simplify the model by setting $F_{\mathrm{grade}} = 0$ throughout the analytical development. To obtain the velocity of the vehicle, the dynamic equation can be arranged in the following form,where the acceleration of the vehicle is obtained as a consequence of interaction between powertrain forces and resistive effects:

$$a_{\mathrm{v}} = \frac{\mathrm{d}v_{\mathrm{v}}}{\mathrm{dt}} = \frac{1}{M_{\mathrm{v}}} \left(\frac{T_{\mathrm{w}}}{r_{\mathrm{w}}} - F_{\mathrm{roll}} - F_{\mathrm{vis}} \right) \tag{3.5}$$

The relationship between wheel force and torque is:

$$F_{\mathrm{w}} = \frac{T_{\mathrm{w}}}{r_{\mathrm{w}}} \tag{3.6}$$

The wheel's angular velocity under ideal conditions without slip, is given by:

$$\omega_{\mathrm{w}} = \frac{v_{\mathrm{ref}}}{r_{\mathrm{w}}} \tag{3.7}$$

T_{w} represents the torque that must be applied to the wheel to allow movement and follow the reference velocity,while taking into account all resistive forces.

3.3 Forward Vehicle Model

The longitudinal dynamical model is implemented in Simulink using a forward approach. This methodology simulates the physical causality of the vehicle motion, where the electric motor force generates vehicle acceleration and velocity. Unlike the backward approach, the actual vehicle response is computed starting from applied Torque. The main equation for the forward model is the eq.(3.5), and the velocity of the vehicle is obtained through integration of this equation

$$v_{\mathrm{v}}(t) = v_{\mathrm{v}}(0) + \int_{0}^{t} a_{\mathrm{v}}(\tau)\,\mathrm{d}\tau \tag{3.8}$$

3.4 Gearbox model

The gearbox allows torque and angular speed conversion between the electric motor and the wheels through a fixed-ratio transmission system. The transmission system is characterized by two main relationships:

1) The angular velocity relationship:

$$\omega_{\mathrm{em}} = \tau_{\mathrm{gb}}\,\omega_{\mathrm{w}} \tag{3.9}$$

2) The torque relationship:

$$T_{\text{em}} = \frac{T_{\text{w}}}{\tau_{\text{gb}}\,\eta_{\text{gb}}^{\text{sgn}\,T_{\text{w}}}} \tag{3.10}$$

where T_{em} is the electric motor torque, η_{gb} the gearbox efficiency and τ_{gb} is the gearbox ratio and the sign function in the efficiency exponent accounts for the direction of the torque.

From eq.(3.10),I can express the wheel torque as:

$$T_{\text{w}} = T_{\text{em}}\,\tau_{\text{gb}}\,\eta_{\text{gb}}^{\text{sgn}\,T_{\text{em}}} \tag{3.11}$$

3.5 Electric Motor Power and Efficiency model

The electric motor power (P_{em}) is defined by the product of angular velocity (ω_{em}) and the torque(T_{em}):

$$P_{\text{em}} = \omega_{\text{em}}\,T_{\text{m}} \tag{3.12}$$

The battery power (P_{b}) links to motor power P_{em} via:

$$P_{\text{b}} = \frac{P_{\text{em}}}{(\eta_{\text{em}}(\omega_{\text{em}}, T_{\text{em}})\,\eta_{\text{inv}})^{\text{sgn}\,P_{\text{em}}}} \tag{3.13}$$

where η_{inv} represents inverter efficiency, which I assume to be ideal ($\eta_{\text{inv}} = 1$) in my study. Given that $\omega_{\text{em}} \geq 0$ the eq. (3.13) can be simplified to:

$$P_{\text{b}} = \frac{P_{\text{em}}}{\eta_{\text{em}}(\omega_{\text{em}}, T_{\text{em}})^{\text{sgn}\,T_{\text{em}}}} \tag{3.14}$$

The motor efficiency, denoted by $\eta_{\text{em}}(\omega_{\text{em}}, T_{\text{em}})$ is determined through an efficiency map that yields the corresponding efficiency value based on the input torque T_{em} and the angular velocity ω_{em}. For simulation purposes,I use a generic motor efficiency map,because the specific efficiency map of the Fiat 500e's motor is not publicly available. This map provides a representative characterization of the typical performance of electric motors in BEV vehicle. The efficiency map utilized in my study is shown in Figure 3.1

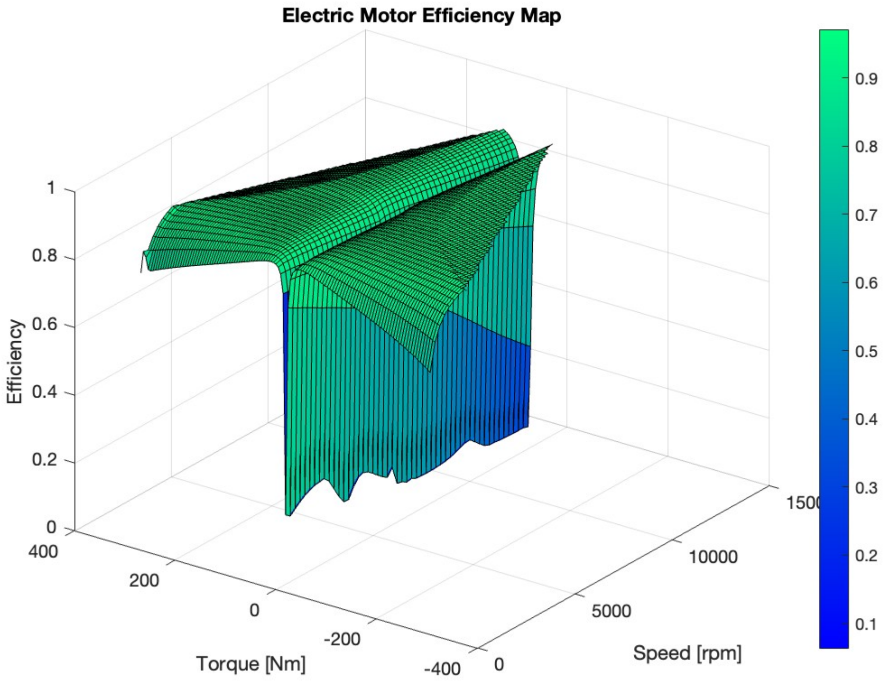

Figure 3.1: Efficiency motor map

3.6 Battery

The battery pack of a Fiat 500e consists of lithium-ion cells connected in series configurations. The most important battery parameters are summarized in Table 3.2.

Using the battery modeling approach presented in [4], the State of Charge

Parameter	Symbol	Value	Unit
Cells in series	N_{s}	108	-
Parallel strings	N_{p}	1	-
Nominal capacity	Q_{nom}	60	Ah
Coulomb efficiency	η_{c}	0.95	-
Battery power converters	η_{b}	0.95	-

Table 3.2: Battery system specifications for the Fiat 500e

(SOC), which is the remaining charge inside a battery, is defined in eq.(3.15),

$$\mathrm{SOC} \equiv \zeta = \frac{Q_{\mathrm{b}}}{Q_{\mathrm{nom}}} \tag{3.15}$$

and the variation of the SOC is defined as

$$\dot{\zeta} = -\frac{I_{\mathrm{b}}}{Q_{\mathrm{nom}}\,\eta_{\mathrm{c}}^{\mathrm{sgn}\,(P_{\mathrm{b}})}} \tag{3.16}$$

where the η_{c} is the Coulomb efficiency and I_{b}, is the battery current that flow out from battery,and it is defined as:

$$I_{\mathrm{b}} = \frac{V_{\mathrm{b}}^{\mathrm{oc}} - \sqrt{V_{\mathrm{b}}^{\mathrm{oc}2} - 4\,R_{\mathrm{b}}^{\mathrm{o}}\,\eta_{b}^{-\,\mathrm{sgn}\;P_{\mathrm{b}}}\,P_{\mathrm{b}}}}{2\,R_{\mathrm{b}}^{\mathrm{o}}} \tag{3.17}$$

where η_{b} is the battery power converters and $V_{\mathrm{b}}^{\mathrm{oc}}$, $R_{\mathrm{b}}^{\mathrm{o}}$ represent, respectively the open circuit voltage and internal resistance of the battery pack,and it is defined as:

$$V_{\mathrm{b}}^{\mathrm{oc}} = N_{\mathrm{s}}\,V_{\mathrm{c}}^{\mathrm{oc}} \tag{3.18}$$

$$R_{\mathrm{b}}^{\mathrm{o}} = N_{\mathrm{s}}\,R_{\mathrm{c}}^{\mathrm{o}} \tag{3.19}$$

where $V_{\mathrm{c}}^{\mathrm{oc}}$ is the ideal open circuit voltage source, that modelled the single cell of the battery, $R_{\mathrm{c}}^{\mathrm{o}}$ is the output resistance of the single cell and N_{s} is the number of cell in series.

The characteristics of single cells $V_{\mathrm{c}}^{\mathrm{oc}}$ and $R_{\mathrm{c}}^{\mathrm{o}}$ depends on their SOC. These crucial pararameters were obtained in laboratory by Professor Novara's doctoral research teal, who provided me the experimental data.

Given that the experimental data consisted of dicrete measurements, it was necessary to develop a continuous representation for implementation within the model framework. The objective was to establish a continuous function that could accurately describe the behavior of the cell throughout the operational range, defined as $\zeta \in (0,1)$.

To address this requirement,I employed MATLAB's [12] interp function. The choice of spline interpolation proved particularly advantageous, as it generated smooth transition that captured the variations in cell behavior. The resulting functions accurately represented $V_{\mathrm{c}}^{\mathrm{oc}}$ and $R_{\mathrm{c}}^{\mathrm{o}}$ across all SOC values, while maintaining the physical behavior of the battery system. Figure 3.2 shows both the experimental data points and the corresponding interpolated curves, allowing a comparison between the discrete measurements and the continuous function obtained by spline interpolation.

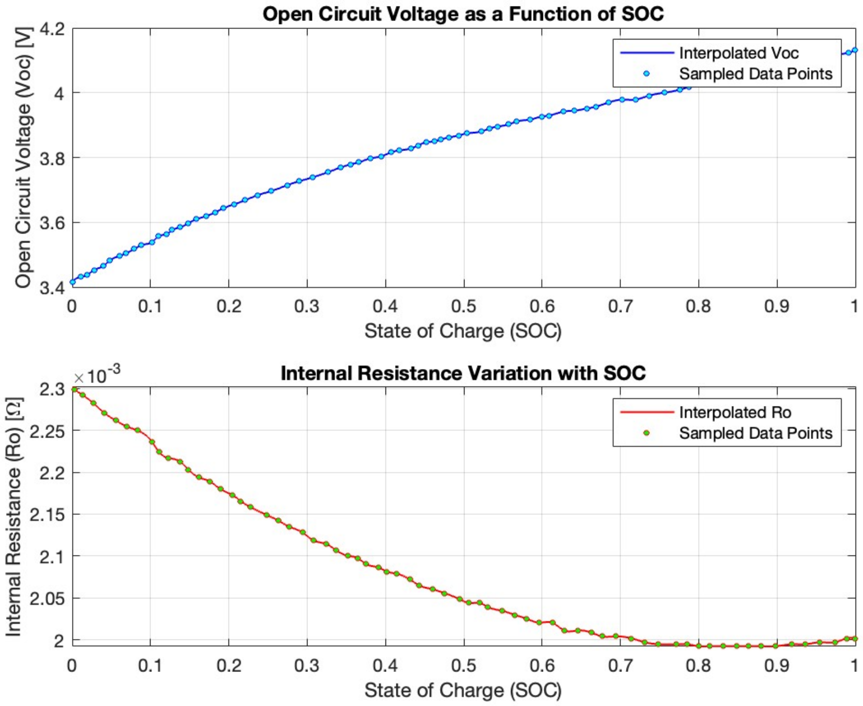

Figure 3.2: Battery cell characteristic parameters as functions of SOC. The upper plot shows the open circuit voltage ($V_{\mathrm{c}}^{\mathrm{oc}}$) exhibiting a monotonic increase with SOC, while the lower plot illustrates the internal resistance ($R_{\mathrm{c}}^{\mathrm{o}}$) displaying an overall decreasing trend.

3.7 CTG

In the context of Adaptive Cruise Control (ACC), the Costant Time Gap (CTG) method is widely used for maintaining a safe distance from the vehicle ahead. Unlike fixed distance strategies, CTG adapts the safe following distance based on the vehicle's current speed. This method aims to hold a constant time interval between vehicles, rather than a constant distance gap.

In my implementation, the CTG controller calculates a target distance between the ego vehicle and the lead vehicle according to the following equation:

$$d_{\text{des}} = d_{\text{min}} + h \cdot v_{\text{current}} \tag{3.20}$$

where d_{des} is the desired distance, d_{min} represents the minimum safety gap when stopped, I set this distance to $0.5\,m$, h is the desired time gap and $v_{current}$ is the ego vehicle's current velocity. To regulate this distance, the controller computes a corrective acceleration based on a PD controller.

$$a_{\text{command}} = k_d \cdot (d_{\text{actual}} - d_{\text{des}}) + k_v \cdot (v_{\text{lead}} - v_{\text{current}}) \tag{3.21}$$

Here, k_d and k_v represent the proportional and derivative gains respectively, where k_d addresses position error, and k_v regulates the relative velocity difference. After trying several values, I settled on $k_d = 2 \cdot \frac{1}{h}$ and $k_v = \frac{1}{h}$ for my experiment setup. The first term corrects distance deviations, while the second one penalizes the difference between vehicles.

The detailed procedure for translating the desired acceleration command into motor torque and the corresponding velocity calculations for the CTG controller are presented in section 5.3.

3.8 State Space Representation

The dynamic of our system can be described through a nonlinear function that characterizes its temporal evolution:

$$\dot{x} = f(x, u) \tag{3.22}$$

This formulation enables us to derive a state-space representation for my continuous-time dynamical system. To formulate this model, let me consider a state vector comprising three variables that characterize our system's behavior:

$$x = \begin{bmatrix} p \\ v \\ \zeta \end{bmatrix} \tag{3.23}$$

where each component represents a physical quantity:

- $x_1 = p$ represent the longitudinal displacement of the vehicle, measured in meters.
- $x_2 = v$ is the instantaneous velocity of the vehicle in meters per second.
- $x_3 = \zeta$ is the battery's state of charge(SOC), which is dimensionless and represents the vehicle's energetic state.

The system's behavior is influenced through a single control input:

$$u = \mathrm{T_{em}} \tag{3.24}$$

which represents the electric motor torque in Newton-meters.

The state-space representation of our system emerges through a careful derivation process.

The first state equation represents position dynamics, and its derivation is straightforward:

$$\dot{x}_1 = \dot{p} = v = x_2 \tag{3.25}$$

The second state equation requires a more detailed derivation process. Starting with eq. (3.5), which describes the vehicle's acceleration:

$$\dot{x}_2 = \dot{v} = \frac{1}{M_\mathrm{v}} \left(\frac{T_\mathrm{w}}{r_\mathrm{w}} - F_\mathrm{roll} - F_\mathrm{vis} \right) \tag{3.26}$$

By substituting the expression for forces from eq. (3.4) expressing T_w through equation (3.11) and replacing v with my state variable x_2, noting that that T_em represents the control input u:

$$\dot{x}_2 = \frac{1}{M_\mathrm{v}} \left(\frac{\tau_\mathrm{gb} \eta_\mathrm{gb}^{u}}{r_\mathrm{w}} u - M_\mathrm{v} g f_0 \operatorname{sgn} x_2 - \frac{1}{2} \rho A_\mathrm{f} C_\mathrm{d} x_2^2 \right) \tag{3.27}$$

The third state equation involves battery dynamics. Starting from the SOC equation (3.16):

$$\dot{x}_3 = \dot{\zeta} = -\frac{I_\mathrm{b}}{Q_\mathrm{nom} \eta_\mathrm{c}^{\operatorname{sgn} u}} \tag{3.28}$$

Here I_b from eq. (3.17), can be rewritten as a function of ζ and P_b:

$$I_\mathrm{b} = \frac{1}{2 R_\mathrm{b}^\mathrm{o}(\zeta)} \left(V_\mathrm{b}^\mathrm{oc}(\zeta) - \sqrt{V_\mathrm{b}^{\mathrm{oc}2}(\zeta) - 4 R_\mathrm{b}^\mathrm{o}(\zeta) \eta_\mathrm{b}^{-\operatorname{sgn} P_\mathrm{b}} P_\mathrm{b}} \right) \tag{3.29}$$

The battery power P_b inside equation (3.29) derives from equation (3.14):

$$P_\mathrm{b} = \frac{\omega_\mathrm{em} u}{\eta_\mathrm{em}(\omega_\mathrm{em}, u)^{\operatorname{sgn} u}} \tag{3.30}$$

The motor angular velocity ω_{em} relates to vehicle velocity through equation (3.9):

$$\omega_{\text{em}} = \frac{\tau_{\text{gb}}}{r_{\text{w}}} x_2 \tag{3.31}$$

This state-space model capture the complete system dynamics. It incorporates mechanical behavior and battery dynamics in a unified mathematical framework. The complete set of differential equations that describe the system evolution can be written as:

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{1}{M_{\text{v}}} \left(\frac{\tau_{\text{gb}} \eta_{\text{gb}}^{u}}{r_{\text{w}}} u - M_{\text{v}} g f_0 \operatorname{sgn} x_2 - \frac{1}{2} \rho A_{\text{f}} C_{\text{d}} x_2^2 \right) \\ -\frac{I_{\text{b}}}{Q_{\text{nom}} \eta_{\text{c}}^{\operatorname{sgn} u}} \end{bmatrix} \tag{3.32}$$

This formulation explicitly showing how the control input affects each state variable.

Chapter 4

Model Predictive Control: From Traditional to Economic Frameworks

Model Predictive Control (MPC) has gained significant attention across various industries in recent years. While it first found widespread use in petrochemical applications, where it remains the dominant control strategy, my interest focuses on its emerging role in automotive systems. What distinguishes MPC from conventional control approaches is its fundamental operating principle: rather than simply reacting to current errors like traditional controllers, MPC actively anticipates system behavior and plans optimal control actions over a prediction horizon.

Unlike classical control methods, widely used in the industrial branch, such as PID or pole placement, which use fixed control laws. MPC takes a different approach. It solves an optimization problem in order to decide the best possible control action. But that's not all, MPC, unlike the methods mentioned above, is able to be more flexible in the control of multiple input multiple output (MIMO) systems, and under certain conditions it guarantees asymptotic stability.

At every time step k, when the system is in state x_k, MPC employs its system dynamics knowledge to analyze possible future trajectories. The controller calculate how the system would change under various input sequences during the next N Steps, where N is the prediction horizon. Upon examination of all solutions. The MPC chooses the best input sequence that best meets our specified optimization criteria within the limits of system constraints.

After computing the optimal input sequence, MPC applies only first input to the system. This single input drive the system to evolve to state x_{k+1}. The process then repeat at the next time step, with the prediction window shifting forward, hence the term "receding horizon control". Figure 4.1 shows how it works.

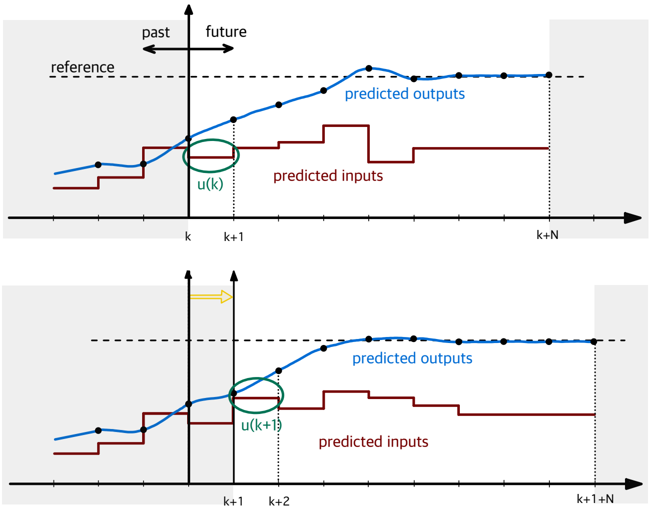

Figure 4.1: Receding Horizon Idea: upper panel shows optimization at time k with implementation of $u(k)$; lower panel illustrates horizon shift to $k+1$ with new input $u(k+1)$. Blue curves track reference trajectory, red steps represent control inputs across horizon N.

This rolling horizon strategy has proven to be remarkably effective in my implementation work. While the controller plans several steps ahead, it maintains flexibility by only committing to immediate actions.

The receding horizon approach offers significant benefits for real world implementation. By only applying the first input of the computed sequence, MPC creates a closed-loop control system. This allows it to adapt to system changes, disturbances and modeling errors.

This step-by-step implementation is crucial. Real systems do not usually perform precisely according to mathematical models. As an example, in following another car, driving at a perfectly constant speed is nearly impossible under real traffic conditions. The lead car may slow down suddenly for a pedestrian, speed up to clear a yellow light, or change its speed due to changing road grades. These natural variations create a dynamic environment that static control strategies struggle to handle. As the system evolves to state x_{k+1}, new measurements provide update information. The controller then recalculates the optimal trajectory based on this fresh information. This helps correct for any deviations from expected behavior.

In my implementation work, I've found this adaptability valuable. The closed-loop nature of MPC means that it continuously re-optimize based on current conditions. It doesn't blindly follow a pre-calculated path. This provides robustness against various uncertainties:

- Model inaccuracies: No mathematical model is perfect.
- Unmeasured disturbances: External factors affect the system.
- Variations in parameters: The characteristics of the system change over time.
- Measurement noise: State information is imperfect.

MPC combines prediction-based planning with continuous feedback correction. This creates a control strategy balancing optimization with adaptability. This makes it suitable for electric vehicles, where conditions constantly change.

The MPC family contains several variants of the algorithm. All the variants share the same fundamental principle of prediction and receding horizon control but differ in the implementation details and capabilities. Nonlinear MPC (NMPC) is the most capable and general class in this family. Unlike simpler linear variants, NMPC can handle complex MIMO systems with nonlinear dynamics.

Note 4.1

In the following sections, I always consider discrete-time systems. This choice stems from the fact that computers, even when operating with very small time steps, work in discrete times. The programs that solve optimization problems require this type of systems (CasaDi,YALMIP,...). For this reason, continuous time systems must be discretized. These discretizations aren't always "good" and don't always faithfully represent the systems. As we'll see later, they sometimes need little tricks to be implemented properly

4.1 Mathematical Basis of NMPC

The most general NMPC framework can be formulated as a finite-horizon optimal control problem. Consider the nonlinear discrete-time system:

$$x_{\mathrm{k}+1} = f(\boldsymbol{x}_{\mathrm{k}}, \boldsymbol{u}_{\mathrm{k}}, \boldsymbol{d}_k) \tag{4.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ represents the system state vector, $\boldsymbol{u} \in \mathbb{R}^m$ are the system inputs, and $\boldsymbol{d}_{\mathrm{k}} \in \mathbb{R}^p$ is the disturbance term.

In my implementation, I focused on deterministic systems where disturbances could be neglected during prediction (though they would naturally occur during actual operation). This simplification leads to the prediction model:

$$x_{\mathrm{k}+1} = f(\boldsymbol{x}_{\mathrm{k}}, \boldsymbol{u}_{\mathrm{k}}) \tag{4.2}$$

We assume the generic nonlinear function $f(\boldsymbol{x}, \boldsymbol{u})$ is continuous and differentiable almost everywhere, which proves sufficient for most practical applications I've encountered. This requirements, while seemingly restrictive, encompass the great majority of systems that are of interest to us while allowing robust numerical solution methods.

4.2 Types of Model Predictive Control

In developing control strategies for electric vehicle, I encountered several MPC variants with distinct characteristics. Each approach offers unique advantages depending on the specific control objectives.

4.2.1 Traditional Nonlinear MPC

Traditional NMPC frameworks typically prioritize reference tracking or regulation problems. These controllers minimize the deviations between system outputs and predefined reference trajectories through cost functions that often take quadratic forms:

$$J = \sum_{i=0}^{N-1} \|x_{\mathrm{i}} - x_{\mathrm{ref}}\|_Q^2 + \|u_{\mathrm{i}} - u_{\mathrm{ref}}\|_R^2 \quad + \quad \|x_{\mathrm{N}} - x_{\mathrm{ref}}\|_P^2 \tag{4.3}$$

where Q,R, and P are weighting matrices that penalize deviations from reference states, inputs and terminal states respectively.

4.2.2 Economic MPC

Economic MPC (EMPC) differs from standard MPC. Basic MPC is built to ensure asymptotic tracking of the reference point (x_{ref}, u_{ref}), without taking into account the costs during the transient phase. The behavior of the closed-loop depends only on the choice of Q and R matrices. A choice of $Q \gg R$ leads to fast tracking. On the other hand a choice of $R \ll Q$ leads to slow tracking.

EMPC takes a different approach. It considers the cost during the transient phase. It doesn't only consider convergence to our reference point, economic optimality is not linked only to convergence. For example, in chemical systems, economic optimality is often characterized by periodic behaviors where the system never converges to a single reference point.

For this reason, EMPC explicitly incorporates economic performance indicators into the cost function. The EMPC cost function takes this form:

$$J = \sum_{i=0}^{N-1} \ell(x_\mathrm{i}, u_\mathrm{i}) + V_o(x_\mathrm{N}) \tag{4.4}$$

where $\ell(x_i, u_i)$ quantifies the actual economic cost of the system operation at state x_i with input u_i and V_o is an offset cost. The terminal cost V_o helps make the system stable by adding extra penalties when system deviates from where we want to end up.

This change from quadratic penalties to economic metrics transforms the controller's behavior. Standard MPC just chases the reference path, in order to arrive at the steady state. In Economic MPC, the controller optimizes the economic cost of the process without referring exclusively to steady state, taking the transient phase into account.This makes it useful not just for following a reference, but also for considering energy usage, which is essential for this thesis. I need to care about vehicle range, not just tracking accuracy.

Note 4.2

Since Economic MPC is a subclass of the more general Nonlinear MPC family, and since I developed my economic cost function step by step (starting from simple reference tracking), I'll present the general MPC formulation in the following sections. I'll point out the specific differences for EMPC when needed - especially concerning things like equilibrium points and the fact that in EMPC we don't have ℓ as positive definite around the equilibrium point. This progression was necessary because the MPC tracking is used as a comparison model for the final formulation of the economic cost function. This developmental approach provided clearer insights into the relative advantages of economic considerations when applied to vehicle control strategies.

4.3 Creating the Mathematical Model for MPC

After examining the different control approaches, I needed to formulate my optimization problem. I structured it in three main components: optimization variables, cost function, and constraints.

4.3.1 Optimization variables

When solving the MPC problem, the controller has to predict the system states for every possible input sequence. That means equations (4.2), that represents how states evolve over time, needs to be embedded into the optimization problem for each step of the prediction horizon.

These state dynamics can be included in two ways: explicit prediction form and implicit prediction form or explicit form.

- **Explicit prediction form**: This approach computes future state offline for a set of possible x_k states. The computed predicted states are expressed as functions of the initial state and predicted input. In this case the optimization variables are only the predicted inputs. This solution is very useful for processors with limited resources, but loses degrees of freedom in finding the optimal solution.

- **Implicit prediction form**: This method considers both predicted inputs and states as optimization variables. The states enter the optimization problem through equality constraints at each time step in the prediction horizon. Including states as optimization variables gives the controller more direct control over system behavior.

For my use case, I chose the implicit prediction form, which means that my predicted vehicle dynamics states - position, velocity and battery state of charge (SOC) - are incorporated into my optimization problem as equality constraints.

The optimization variables will therefore include both the predicted states and inputs. For a generic time instant k, I denote them as follows:

$$\{x_{i|k}\}_{i=0}^{N}, \qquad \{u_{i|k}\}_{i=0}^{N-1} \tag{4.5}$$

where N is the prediction horizon.

The notation "$t|k$" has the following meaning: i represents the time instant of the optimization variable along the prediction horizon $i \in [0,1,...,N]$, while k indicates the time instant where the prediction starts. To understand better, More specifically, at time k when my system is at state x_k, I begin predicting future states to calculate the input u_k, needed to progress to state x_{k+1}.

Note 4.3

At the beginning of each prediction window, $x_{0|k} = x_k$.

The predicted states and predicted inputs sets are defined as follows:

$$\boldsymbol{X}_k = \{x_{0|k}, x_{1|k}, ..., x_{N|k}\} \tag{4.6a}$$

$$\boldsymbol{U}_k = \{u_{0|k}, u_{1|k}, ..., u_{N-1|k}\} \tag{4.6b}$$

4.3.2 Cost Function

The general expression of the cost function for a nonlinear MPC takes this form:

$$J_{\text{NMPC}}(\boldsymbol{X}_{\text{k}}, \boldsymbol{U}_k) = J_{[0,N-1]}(\boldsymbol{X}_{\text{k}}, \boldsymbol{U}_k) + V(x_{N|k}) \equiv \sum_{i=0}^{N-1} \ell(x_{i|k}, u_{i|k}) + V(x_{N|k}) \tag{4.7}$$

Here, J_{NMPC} represent the cost function, ℓ is the stage cost function, that captures the performance at each prediction step, while and V is the terminal cost function that penalizes the deviations at the end of the prediction horizon. The term $J_{[0,N-1]}$

represents the nonlinear MPC cost function without terminal cost.

Note 4.4

V, the terminal cost, is not required in the cost function definition. However, it is request to ensure asymptotic stability(4.4.2)

My first cost function, described in equation (4.23b), aims to follow the preceding vehicle. This means the optimization problem must provide me with an input sequence that reaches the desired state, potentially as $k \to \infty$

For the NMPC to perform tracking effectively, certain initial conditions must be met:

- There must exist an input u_r such that the pair (x_r, u_r) is an equilibrium point for the open-loop system. This means there must be a u_r that maintains the system at the equilibrium point. If this input does not exist, keeping the system at that point would be impossible. In mathematical terms:

$$x_r = f(x_r, u_r) \tag{4.8}$$

- The cost function must penalize the distance between the current system state x and the equilibrium point x_r Adding a penalty on input u is not required to prove stability but can help to limit control signal effort by seeking the smallest possible u. In case the input doesn't have a reference, or when the input at equilibrium state isn't easily determined, the penalization can be done with respect to the point $(\boldsymbol{x}_r,0)$. In this scenario, the stage function ℓ must be positive definite in a a neighborhood of $(\boldsymbol{x}_r,0)$. This approach penalizes the distance of U from the origin while still ensuring state tracking to x_r. This method is used, for example, when dealing with systems where the steady-state input is difficault to compute analytically.

To ensure our stage cost function ℓ penalizes the distance from our equilibrium point for all $\boldsymbol{x} \in \mathbb{R}^n$, ℓ must be positive definite in a neighborhood of the equilibrium point $(\boldsymbol{x}_r, \boldsymbol{u}_r)$:

$$\ell(\boldsymbol{x}, \boldsymbol{u}) = \begin{cases} 0 & \text{if } (\boldsymbol{x}, \boldsymbol{u}) = (\boldsymbol{x}_r, \boldsymbol{u}_r) \\ > 0 & \forall (\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^n \times \mathbb{R}^m \setminus \{(\boldsymbol{x}_r, \boldsymbol{u}_r)\} \end{cases} \tag{4.9}$$

The theoretical considerations established for the constant equilibrium point $(\boldsymbol{x}_r, \boldsymbol{u}_r)$ can be extended, with appropriate reformulations, to the tracking of time-varying trajectories $\boldsymbol{x}_r(k)$, which represent our reference trajectory. This extension allows us to address more dynamic control scenarios where the target state evolves over time, as in the case of sinusoidal reference velocity or WLTP cycle.

A typical choice for the cost function in NMPC formulation, which also corresponds to my first cost function implementation without terminal cost, takes the following form:

$$J(\boldsymbol{X}_k, \boldsymbol{U}_k) = \sum_{i=0}^{N-1} \ell(x_{\mathrm{i|k}}, u_{\mathrm{i|k}}) + V(x_{N|k}) = \sum_{i=0}^{N-1} (x_{\mathrm{i|k}} - \boldsymbol{x}_r)^T \boldsymbol{Q} (x_{\mathrm{i|k}} - \boldsymbol{x}_r) + u_{\mathrm{i|k}}^T \boldsymbol{R} u_{\mathrm{i|k}} \tag{4.10}$$

where Q and R are positive definite diagonal matrices.

Note 4.5

Q can be positive semi-definite if the pair $(Q^{1/2}, A)$ is observable.

> **Def. (Observability):** is the property of a system that enables the estimation of its initial state through the measurement of output $y(\cdot)$ and input $u(\cdot)$ over a specified time interval.

In other words, the initial state of the system $x(0)$ can be determined if we know the input-output behavior over a finite time horizon.

This cost function satisfies the condition in eq. (4.9). The diagonal matrices $\boldsymbol{Q}$

and $\boldsymbol{R}$ represent our weights, which allow us to adjust the trade-off between the rate of convergence to the equilibrium point and how much effort is required to achieve this equilibrium state. Larger values of $\boldsymbol{Q}$, giving us a faster system response to reach $\boldsymbol{x}_r$. Instead, larger values of $\boldsymbol{R}$, penalize the amplitude of the control signal, resulting in slower convergence to the equilibrium point.

Note 4.6

For the economic MPC, we need to move away from the positive definiteness around a predefined equilibrium (x_r, u_r). The a-priori assumption of equilibrium loses meaning in this context. Instead, equilibrium takes on a different interpretation. The equilibrium sought by the MPC balances the stage cost and system dynamics. Typically, ℓ is not positive definite in this context. This necessitates a different definition:

Def. (Optimal equilibrium): An equilibrium (x^e, u^e) is called an optimal equilibrium if it yields the lowest value of the cost function among all admissible equilibria [9]

$$\ell(x^e, u^e) \leq \ell(x, u) \quad \forall (x, u) \in \mathcal{X} \times \mathcal{U} \text{ with } f(x, u) = x \tag{4.11}$$

Unlike traditional MPC where the stage cost penalizes distance from a known equilibrium point, Economic MPC uses a stage cost containing economic performance metrics that might not reach their minimum at the desired equilibrium state. This fundamental difference shifts focus to optimizing operational cost during both steady state and transient phases. But losing positive definiteness of ℓ creates the need to introduce dissipativity concepts, as we'll see in the asymptotic stability section. Dissipativity provides a more general energy-based framework for analyzing system behavior.

4.3.3 Constraints

As previously stated, one of the characteristics that make MPC advantageous compared to other types of control is the ability to incorporate constraints directly into the calculation of optimal solution.

According to Goodwin et al. [8], there exist four fundamental approaches to constraint handling in control problems:

- **Cautious approach:** This strategy is implemented by ensuring that constraint are never violated under any circumstances. The controller behaves conservatively, maintaining a significant margin from constraint boundaries. This affects the performance of the controllers, typically resulting in extremely slow convergence to equilibrium states.

- **Serendipitous approach:** This method initially ignores constraints in the construction of the controller, developing the controller as if constraints did not exist. Constraints are subsequently imposed on the completed design. For instance, with input constraints, the controller is first designed without restrictions, then the input is limited according to constraint. This type of approach allow occasional constraint violation.

- **Evolutionary approach:** This strategy employs an iterative trial-and-error methodology. The controller is initially constructed without constraint implementation, then progressively modified based on observed constraint violations during operation.

- **Tactical approach:** this is the most sophisticated strategy, as well as the strategy adopted by MPC, where constraints are incorporated directly into the controller formulation. They are taken into account from the beginning. By considering constraints from the outset, the optimizer that minimizes the cost function seeks the best solution while inherently respecting the imposed constraints.

MPC constraints get added to the minimization problem in this format:

$$\boldsymbol{x}_{i|k} \in \mathcal{X}, \quad i = 0{,}1, \ldots, N \tag{4.12a}$$

$$\boldsymbol{u}_{i|k} \in \mathcal{U}, \quad i = 0, 1, \ldots, N-1 \tag{4.12b}$$

$$\boldsymbol{x}_{N|k} \in \mathcal{X}_F \tag{4.12c}$$

The first equations restricts the predicted states to remain within an admissible region, $\mathcal{X} \subset \mathbb{R}^n$, the second one restrict control input into feasible set $\mathcal{U} \subset \mathbb{R}^m$. The third equation represents the terminal constraint applied to the final prediction state, $\mathcal{X}_F \subset \mathbb{R}^n$.

MPC constraints can be insert inside minimization problem can be categorized into two main types: equality constraint and inequality constraint.

Equality Constraint

Equality constraints enforce exact relationship and typically take the form of:

$$c(\boldsymbol{x}i|k, \boldsymbol{u}i|k) = 0 \tag{4.13}$$

When dealing with a constant and we want to set $c(x_{i|k}, u_{i|k})$ equal to c_{ref}, we prefer to write it as $c(x_{i|k}, u_{i|k}) - c_{\text{ref}} = 0$. This reformulation makes implementation inside solvers easier, helping us write constraints in matrix form. So usually, we write:

$$0 \leq c(x_{i|k}, u_{i|k}) - c_{\text{ref}} \leq 0 \tag{4.14}$$

The most important equality constraints used in MPC are the equations describing system dynamics. They ensure the system evolves according to the system model. Also the terminal constraint is usually in this form.

In my implementation, I use a terminal equality constraint:

$$x_{2,N|k} - x_{\text{ref}} = 0 \tag{4.15}$$

This makes the vehicle velocity at the prediction horizon's end match exactly the reference value. The terminal constraint is important for asymptotic stability and recursive feasibility, but not mandatory in MPC problem formulation.

Inequality constraints

Inequality constraints define boundaries rather than exact values, typically represented as:

$$c(\boldsymbol{x}_{i|k}) \leq 0, \quad i = 0, 1, \ldots, N \tag{4.16a}$$

$$c(\boldsymbol{u}_{i|k}) \leq 0, \quad i = 0, 1, \ldots, N-1 \tag{4.16b}$$

For my application, I constraint the motor torque, my control input, between minimum and maximum allowable values:

$$T_{\text{em,min}} \leq u_{i|k} \leq T_{\text{em,max}}, \quad i = 0, 1, \ldots, N-1 \tag{4.17}$$

For the state, I constraint velocity and SOC, respectively x_2 and x_3 such that:

$$v_{\text{veh},min} \leq x_{2,i|k} \leq v_{\text{veh},max}, \qquad i = 0, 1, \ldots, N \tag{4.18a}$$

$$\zeta_{min} \leq x_{3,i|k} \leq \zeta_{max}, \qquad i = 0, 1, \ldots, N \tag{4.18b}$$

For clarity, I listed the specific numerical values of the constraints used in my implementation in Table 4.1.

The negative value for minimum velocity allows the vehicle to move in reverse

Parameter	**Symbol**	**Value**	**Unit**
Maximum motor torque	$T\text{em, max}$	280	$N \cdot m$
Minimum motor torque	$T\text{em, min}$	-280	$N \cdot m$
Maximum vehicle velocity	$v\text{veh, max}$	50	m/s
Minimum vehicle velocity	$v\text{veh, min}$	-5	m/s
Maximum battery SOC	ζmax	1	-
Minimum battery SOC	ζ_{min}	0.01	-

Table 4.1: Constraint parameter values used in the MPC formulation

when necessary, through in typical driving scenarios this capability is not utilized. Regarding the battery SOC, I put 0.01 to minimum value to avoid numerical problem around 0.

Having these two classes of constraints, I can rewrite them in the following form:

$$lbd_j \le c_{\text{ineq,j}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \le ubd_j, \quad j = 1, \dots, p \tag{4.19a}$$

$$c_{\text{eq,l}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) = 0, \quad l = 1, \dots, q \tag{4.19b}$$

$$lbd_j \le c_{\text{ineq,j}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \le \qquad ubd_j, \quad j = 1, \dots, p \tag{4.20a}$$

$$0 \le c_{\text{eq,l}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \le \qquad 0, \quad l = 1, \dots, q \tag{4.20b}$$

$$\begin{bmatrix} lbd_1 \\ \vdots \\ lbd_p \\ 0 \\ \vdots \\ 0 \end{bmatrix} \le \begin{bmatrix} c_{\text{ineq,1}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \\ \vdots \\ c_{\text{ineq,p}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \\ c_{\text{eq,1}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \\ \vdots \\ c_{\text{eq,q}}(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \end{bmatrix} \le \begin{bmatrix} ubd_1 \\ \vdots \\ ubd_p \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.21}$$

$$LBD \le G \le UBD \tag{4.22}$$

where :

- $LBD = [lbd_1, \cdots, lbd_p, \mathbf{0}_q]^T$ is the vector of lower bounds.
- $UBD = [ubd_1, \cdots, ubd_p, \mathbf{0}_q]^T$ is the vector of upper bounds.
- G represents the composite vector of inequality and equality constraints.

4.3.4 Full Mathematical Formulation of MPC

Collecting all information from the previous sections, I can write the MPC problem formulation as an optimization problem, specifically a "finite horizon constrained optimal control problem". For the cost function, I'll include the most complete version, which is also my first cost function used in simulations. As mentioned earlier, this cost function contains both a stage cost term h and a terminal cost V.

MPC optimization problem

$$(\boldsymbol{X}_k^*, \boldsymbol{U}_k^*) = \arg \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} J(\boldsymbol{X}_k, \boldsymbol{U}_k) \tag{4.23a}$$

$$\begin{aligned} J(\boldsymbol{X}_k, \boldsymbol{U}_k) &= J_{[0,N-1]}(\boldsymbol{X}_k, \boldsymbol{U}_k) + V(\boldsymbol{x}_{N|k}) \\ &= \sum_{i=0}^{N-1} \ell(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \; + V(\boldsymbol{x}_{N|k}) \\ &= \sum_{i=0}^{N-1} (x_{\mathrm{i|k}} - \boldsymbol{x}_r)^T \boldsymbol{Q} (x_{\mathrm{i|k}} - \boldsymbol{x}_r) + u_{\mathrm{i|k}}^T \boldsymbol{R} u_{\mathrm{i|k}} + (x_{\mathrm{N|k}} - \boldsymbol{x}_r)^T \boldsymbol{P} (x_{\mathrm{N|k}} - \boldsymbol{x}_r) \end{aligned} \tag{4.23b}$$

subject to:

$$x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad i = 0,1,\ldots,N-1 \tag{4.23c}$$

$$x_{0|k} = x_k \tag{4.23d}$$

$$x_{i|k} \in \mathcal{X} \quad i = 0,1,\ldots,N \tag{4.23e}$$

$$u_{i|k} \in \mathcal{U} \quad i = 0,1,\ldots,N-1 \tag{4.23f}$$

$$x_{N|k} \in \mathcal{X}_F \tag{4.23g}$$

$$(*) \tag{4.23h}$$

Where the optimization variables are the same as defined in section 4.3.1, namely

$$\boldsymbol{X}_k = \{\boldsymbol{x}_{0|k}, \boldsymbol{x}_{1|k}, \dots, \boldsymbol{x}_{N|k}\} \tag{4.24}$$

$$\boldsymbol{U}_k = \{\boldsymbol{u}_{0|k}, \boldsymbol{u}_{1|k}, \dots, \boldsymbol{u}_{N-1|k}\} \tag{4.25}$$

These sets represent, respectively, the predicted state trajectory and the predicted input sequence over the optimization horizon. The asterisk present in the problem at (4.23h) indicates additional constraints that can be placed there for supplementary conditions that might be incorporated according to specific control objectives or system requirements. The basic version of the problem, as previously mentioned, does not include the terminal cost function V and the terminal constraint (4.23g), and does not contain additional constraints:

$$(\boldsymbol{X}_k^*, \boldsymbol{U}_k^*) = \arg \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} J(\boldsymbol{X}_k, \boldsymbol{U}_k)$$

$$J(\boldsymbol{X}_k, \boldsymbol{U}_k) = J_{[0,N-1]}(\boldsymbol{X}_k, \boldsymbol{U}_k) = \sum_{i=0}^{N-1} \ell(\boldsymbol{x}_{i|k}, \boldsymbol{u}_{i|k}) \tag{4.26a}$$

subject to:

$$x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad i = 0{,}1, \dots, N-1 \tag{4.26b}$$

$$x_{0|k} = x_k \tag{4.26c}$$

$$x_{i|k} \in \mathcal{X} \quad i = 0{,}1, \dots, N \tag{4.26d}$$

$$u_{i|k} \in \mathcal{U} \quad i = 0{,}1, \dots, N-1 \tag{4.26e}$$

$\boldsymbol{X}_k^*$ and $\boldsymbol{U}_k^*$ represent the optimal predicted trajectory and the optimal predicted input sequence, respectively, obtained from solving the minimization problem:

$$\boldsymbol{X}_k^* = \{x_{0|k}^*, \ldots, x_{N|k}^*\} \tag{4.27a}$$

$$\boldsymbol{U}_k^* = \{u_{0|k}^*, \ldots, u_{N-1|k}^*\} \tag{4.27b}$$

The optimal value of the cost function is denoted as J_k^* or, expressed as a function of the initial state, $J^*(\boldsymbol{x}_k)$.

Returning to the receding horizon principle of model predictive control, as explained at the beginning of this chapter and illustrated in Figure 4.1, only the first input of the optimal control sequence is applied to the system. At the k-th instant of our closed-loop cycle, we can therefore write:

$$\mathbf{X} = \{x_0, x_1, \ldots, x_k\} \tag{4.28a}$$

$$\mathbf{U} = \{u_{0|0}^*, u_{0|1}^*, \ldots, u_{0|k}^*\} \tag{4.28b}$$

The state x_k is calculated by applying the input $u_{0|k-1}^*$. In an ideal case where there are no discrepancies between the first predicted state $x_{1|k-1}^*$ and the actual state x_k, we could write the equality $x_k = x_{1|k-1}^*$. However, this relationship generally does not hold due to measurement errors, model-system discrepancies, etc.

For notational simplicity, we denote $u_k = u_{0|k}^*$. Thus, the set of inputs at the k-th instant becomes:

$$\mathbf{U} = \{u_0, u_1, \ldots, u_k\} \tag{4.29}$$

Note 4.7

In practical applications, rather than in idealized scenarios, the control cycle has a finite temporal duration T_{tot}, which corresponds to the time instant at which the cycle terminates. Consequently, the previously defined sets become:

$$\mathbf{X} = \{x_0, x_1, \ldots, x_k, \ldots, x_{\text{T}_{\text{tot}}}\} \tag{4.30a}$$

$$\mathbf{U} = \{u_0, u_1, \ldots, u_k, \ldots, u_{\text{T}_{\text{tot}}-1}\} \tag{4.30b}$$

4.4 Asymptotic Stability and Recursive Feasibility

After presenting the complete mathematical formulation of general MPC, I will now introduce fundamental theorems useful for proving the stability of Economic MPC controller. While recursive feasibility analysis applies broadly across the nonlinear MPC family, the asymptotic stability properties demand distinct treatment for economic formulations. The absence of positive definiteness in the stage cost ℓ precludes conventional Lyapunov-based stability analysis typically employed for standard nonlinear MPC. Instead, we must introduce the concept of dissipativity For our controller to be stable, it must be recursively feasible.

4.4.1 Recursive Feasibility

A MPC optimization problem is called **recursively feasible** if, for every initial state x_0 where a feasible solution exists at time $k = 0$, the optimization problem remains feasible for all subsequent time instants $k \geq 1$ under the application of the control law obtained from MPC into the closed-loop trajectory.

Sometimes, even if the problem is feasible for $k = 0$, it becomes infeasible for other time instants $k \neq 0$. This property makes sure the controller doesn't enter a region where satisfying constraints becomes impossible, thus preventing the controller from calculating a valid control action for state x_k.

It is important to note that the set of solutions that guarantee a feasible closed-loop trajectory is a subset of the states for which there exists at least one feasible open-loop solution [13].Open-loop feasibility only ensures that a solution exists for the current prediction horizon, without guaranteeing that the problem remains feasible in subsequent steps under MPC control law application.

Note 4.8

Recursive feasibility does not imply stability of the closed-loop systems

In order to prove recursive feasibility I need to introduce the concept of invariant set:

Def. (Invariant Set): A set $\mathcal{O}$ is called positively invariant for system $x_{k+1} = f_\kappa(x(k))$, if

$$x(k) \in \mathcal{O} \Rightarrow f_\kappa(x(k)) \in \mathcal{O},\ \forall k \in \mathbb{N}$$

This means, that if I start from a point inside the set $\mathcal{O}$, my trajectory remains inside $\mathcal{O}$ for all future instants of time.

Ensuring recursive feasibility means that the set of initial states is a control invariant set for the system described by the equation (4.2). This set coincides with the set of states that lead to a feasible close-loop trajectory.

Assumption 4.4.1

The terminal set $\mathcal{X}_F$ of (4.23g) is a control invariant set for the system (4.2) if, for $x_k \in \mathcal{X}_F$, it always exist an input $\mathbf{u}'$ such that:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}') \in \mathcal{X}_F \tag{4.31}$$

Initially, in eq. (4.8) we set as initial conditions for our MPC that x_r is an equilibrium point for our system. From this fact, we can determine that the smallest control invariant set containing x_r is simply the singleton set:

$$\mathcal{X}_F = \{x_r\} \tag{4.32}$$

This makes sense because having $u' = u_r$ ensures that $f(x_r, u') = x_r \in \mathcal{X}_F$. In other words, the system stays at the equilibrium point when the corresponding equilibrium input is applied.

Theorem: Recursive Feasibility of MPC

Consider the following general MPC problem:

$$(\boldsymbol{X}_k^*, \boldsymbol{U}_k^*) = \arg \min_{\boldsymbol{X}_k, \boldsymbol{U}_k} J(\boldsymbol{X}_k, \boldsymbol{U}_k)$$

subject to:

$$x_{i+1|k} = f(x_{i|k}, u_{i|k}), \quad i = 0,1,\ldots,N-1 \tag{4.33a}$$

$$x_{0|k} = x_k \tag{4.33b}$$

$$x_{i|k} \in \mathcal{X} \quad i = 0,1,\ldots,N \tag{4.33c}$$

$$u_{i|k} \in \mathcal{U} \quad i = 0,1,\ldots,N-1 \tag{4.33d}$$

$$x_N \in \mathcal{X}_F \tag{4.33e}$$

with a terminal constraint set $\mathcal{X}_F = \{x_r\}$ where x_r is the equilibrium point of the system. If:

- $\mathcal{X}$ and $\mathcal{U}$ are compact sets containing x_r and u_r, respectively
- The function $f(x, u)$ is continuous
- The optimization problem is feasible for the initial state x_0

Then the MPC optimization problem remains feasible for all subsequent time steps, ensuring recursive feasibility of the closed-loop system [9] [13]

Proof of Recursive Feasibility for MPC with Terminal Constraint

Consider the MPC problem formulated in (4.33) with a terminal constraint set $\mathcal{X}_F = \{x_r\}$ where x_r is the equilibrium point of the system.

To establish recursive feasibility, i use an inductive approach, demonstrating that feasibility at any time step k ensures feasibility at the subsequent time step k+1.

Starting from the third assumption, the MPC optimization problem is feasible at time $k = 0$. Therefore, there exists an optimal control sequence and the corresponding trajectory of the predicted states.

$$\begin{aligned} U_0^* &= (u_{0|0}^*, u_{1|0}^*, \ldots, u_{N-1|0}^*) \\ X_0^* &= (x_{0|0}^*, x_{1|0}^*, \ldots, x_{N|0}^*) \end{aligned} \tag{4.34}$$

where $x_{0|0}^* = x_0$ and $x_{N|0}^* = x_r$ due to the terminal constraint (4.33e).

Suppose that at an arbitrary step $k \geq 0$, the MPC problem admits a feasible solution, this means that there exists an optimal control sequence and the corresponding state trajectory :

$$\begin{aligned} U_k^* &= (u_{0|k}^*, u_{1|k}^*, \ldots, u_{N-1|k}^*) \\ X_k^* &= (x_{0|k}^*, x_{1|k}^*, \ldots, x_{N|k}^*) \end{aligned} \tag{4.35}$$

The terminal constraint (4.33e) ensures that $x_{N|k}^* = x_r$. Since x_r is an equilibrium point, by definition(as established in equation (4.8)), there exists an input u_r such that:

$$f(x_r, u_r) = x_r \tag{4.36}$$

According to the receding horizon principle, presented at the begin of this chapter 4, only the first control input $u_k = u_{0|k}^*$ is applied to the system. Consequently, the state at time $k+1$ becomes:

$$x_{k+1} = f(x_k, u_k) = f(x_{0|k}^*, u_{0|k}^*) = x_{1|k}^* \tag{4.37}$$

At time $k+1$, i can construct a candidate control sequence:

$$\hat{U}_{k+1} = (u_{1|k}^*, u_{2|k}^*, \ldots, u_{N-1|k}^*, u_r) \tag{4.38}$$

This sequence, when applied to the system starting from the state $x_{k+1} = x^*_{1|k}$, produces the predicted state trajectory:

$$\hat{X}_{k+1} = (x^*_{1|k}, x^*_{2|k}, \ldots, x^*_{N|k}, f(x^*_{N|k}, u_r)) \tag{4.39}$$

Given that $x^*_{N|k} = x_r$ and $f(x_r, u_r) = x_r$, this trajectory simplifies to:

$$\hat{X}_{k+1} = (x^*_{1|k}, x^*_{2|k}, \ldots, x_r, x_r) \tag{4.40}$$

Checking if the candidate solution satisfy all the constraint in the optimization problem, i have:

- **Dynamic constraint:** By construction the state trajectory $\hat{X}_{k+1}$ follow the system dynamics under the input sequence $\hat{U}_{k+1}$, satisfying the equation:

$$x_{i+1|k+1} = f(x_{i|k+1}, u_{i|k+1}),\ \forall i = 0{,}1, \ldots, N-1$$

- **Initial state constraint:** The first element of the candidate trajectory is $x_{0|k+1} = x^*_{1|k} = x_{k+1}$.

- **State constraints:** Since the original solution (X^*_k, U^*_k) is feasible, i have that $x^*_{i|k} \in \mathcal{X}$ for $i = 1, 2, \ldots, N$. Therefore, $x_{i|k+1} = x^*_{i+1|k} \in \mathcal{X}$ for $i = 0, 1, \ldots, N-1$. Additionally, $x_{N|k+1} = x_r \in \mathcal{X}$ by my assumption that $\mathcal{X}$ is a compact set containing x_r. Thus, all state constraints are satisfied.

- **Input constraints**: Similarly, since (X^*_k, U^*_k) was feasible, $u^*_{i|k} \in \mathcal{U}$ for $i = 1, 2, \ldots, N-1$. Therefore, $u_{i|k+1} = u^*_{i+1|k} \in \mathcal{U}$ for $i = 0, 1, \ldots, N-2$. Moreover, $u_{N-1|k+1} = u_r \in \mathcal{U}$ by assumption. Hence, all input constraints are also satisfied.

- **Terminal constraint**: The final state in the candidate trajectory is $x_{N|k+1} = x_r$, which belongs to $\mathcal{X}_F = \{x_r\}$, thus satisfying the terminal constraint

(4.33e).

Having verified that all constraints are satisfied, we conclude that the candidate solution $(\hat{X}_{k+1}, \hat{U}_{k+1})$ is feasible for the MPC problem at time $k+1$. While this solution may not be optimal, its existence guarantees the feasibility of the optimization problem at time $k+1$.

In conclusion, starting from the assumption that the MPC is feasible at the initial time and as previously demonstrated, it is feasible at time k, by induction it is feasible at time k+1. I can conclude that my problem with the terminal constraint remains feasible for all subsequences k>= 0, this guarantees the Recursive feasibility of the close-loop system.

The compactness of $\mathcal{X}$ and $\mathcal{U}$ and the continuity of $f(x,u)$ serve as foundational theoretical conditions that, although not directly employed in our proof, ensure the well-posedness of the optimization problem at each step. These mathematical properties, even if not used within the proof, guarantee that the set of solutions is bounded and non-empty throughout the process.

4.4.2 Asymptotic stability

Standard MPC, like the reference tracking approach used in my first cost function can prove stability through Lyapunov theory. Lyapunov proposed a method to show system stability without finding the actual system trajectory. Instead, it uses a generalized energy function.

The basic idea behind Lyapunov stability analysis is simple: if we can find a positive definite function that decrease along the system trajectories, the system will reach the equilibrium.

However the assumption of monotonically decreasing cost function isn't valid for all nonlinear systems and for all cost function, and in particular is not valid for the cost function related to the general economic MPC.

In order to prove the stability of Economic MPC , i need to introduce the concept of dissipativity:

Def. (Dissipativity [2]): A control system $f(x, u)$ is dissipative with respect to a supply rate s: $\mathcal{X} \times \mathcal{U} \to \mathbb{R}$ if there exist a function λ: $\mathcal{X} \to \mathbb{R}$ such that

$$\lambda(f(x,u)) - \lambda(x) \leq s(x,u), \quad \forall (x,u) \in \mathcal{X} \times \mathcal{U}$$

If in addition $\rho : \mathcal{X} \to \mathbb{R}_{\geq 0}$ positive definite exists such that

$$\lambda(f(x,u)) - \lambda(x) \leq -\rho(x) + s(x,u),$$

then the system is said to be strictly dissipative

Interpreting this definition in physical terms, the element can be viewed as:

- $\lambda(f(x,u))$: energy stored at time k+1.
- $\lambda(x)$: energy stored at time k
- $s(x,u)$: externally supplied energy

A physical system cannot create energy, it can only dissipate it. And only a system capable of dissipating energy can converge to a stable equilibrium point.

If we define the externally supplied energy as:

$$s(x,u) = \ell(x,u) - \ell(x^e, u^e)$$

where $\ell(x,u)$ is the stage cost function and $\ell(x^e, u^e)$ is the value of this cost function in the equilibrium point, we can interpret dissipation as the system's tendency to dissipate excess energy relative to the minimum equilibrium cost.

This mathematical definition corresponds to the intuitive idea that physical systems naturally tend toward states of minimum energy.

Knowing the definition of dissipativity, we can look up the function $\lambda : \mathcal{X} \to \mathbb{R}$ such that:

$$\min_{x\in\mathcal{X},u\in\mathcal{U}} \ \ell(x,u) + \lambda(x) - \lambda(f(x,u)) \geq \ell(x^e,u^e)$$

If we define the *rotated stage cost function* as:

$$L(x,u) = \ell(x,u) + \lambda(x) - \lambda(f(x,u))$$

we obtain:

$$\min_{x\in\mathcal{X},u\in\mathcal{U}} \ L(x,u) \geq \ell(x^e,u^e) = L(x^e,u^e)$$

This reformulation possesses interesting properties: the rotated cost function $L(x,u)$ reaches its minimum values under the same constraints as the economic MPC at the equilibrium point (x^e,u^e), just like $\ell(x,u)$. Thus, even though we use different cost functions, we obtain the same minimum.

This function has properties that make it more suitable for stability analysis. To proceed, we need to define also the *rotated terminal cost*

$$\tilde{V}_f(x) = V_f(x) + \lambda(x) - V_f(x^e) - \lambda(x^e) \tag{4.41}$$

Theorem: Asymptotic Stability of EMPC with Terminal Constraints

Consider the following Economic MPC [6] [9]:

$$\min_{u} J_{[0,N-1]}(\boldsymbol{X}_k, \boldsymbol{U}_k) + V_f(\boldsymbol{x}_{N|k}) = \sum_{i=0}^{N-1} \ell(x(i|k), u(i|k)) + V_f(x_{N|k})$$

subject to

$$\begin{aligned} x_{i+1|k} &= f(x_{i|k}, u_{i|k}), \quad i = 0{,}1, \ldots, N-1 \\ x_{0|k} &= x_k \\ x_{i|k} &\in \mathcal{X} \quad i = 0{,}1, \ldots, N \\ u_{i|k} &\in \mathcal{U} \quad i = 0{,}1, \ldots, N-1 \\ x_{N|k} &\in \mathcal{X}_f \end{aligned}$$

Assume that

1. The system is strictly dissipative at the equilibrium point $(x^e, u^e) \in \mathcal{X} \times \mathcal{U}$ with a storage function: $\lambda : \mathcal{X} \to \mathbb{R}$ bounded from below with $\lambda(x^e) = 0$ i.e, there exist $\rho \in \mathcal{K}_\infty$ such that:

$$\ell(x,u) - \ell(x^e,u^e) + \lambda(x) - \lambda(f(x,u)) \geq \rho(||x - x^e||),\ \forall (x,u) \in \mathcal{X} \times \mathcal{U}$$

2. The terminal region $\mathcal{X}_f \subseteq \mathcal{X}$ is compact and contains x^e in its interior;
3. There exists a control law $\kappa_f : X_f \to U$ such that:

$$V_f(f(x, \kappa_f(x))) = V_f(x) - \ell(x, \kappa_f(x)) + \ell(x^e, u^e),\ \forall x \in \mathcal{X}_f$$

   with $V_f(x^e) = 0$;
4. The optimization problem is recursively feasible.

Then, the equilibrium point x^e is asymptotically stable for the closed-loop system under the economic MPC control law.

Proof: We construct a rotated terminal cost corresponding to our rotated stage cost:

$$\tilde{V}_f(x) = V_f(x) + \lambda(x)$$

since $V_f(x^e) = \lambda(x^e) = 0$

If the pair $(V_f(\cdot), \ell(\cdot))$ satisfies Assumption 3, then the pair $(\tilde{V}_f(\cdot), L(\cdot))$ satisfies the following property:

$$\tilde{V}_f(f(x, \kappa_f(x))) - \tilde{V}_f(x) = -L(x, \kappa_f(x)) \quad \forall x \in X_f$$

$$\begin{aligned}
\tilde{V}_f(f(x, \kappa_f(x))) - \tilde{V}_f(x) &= V_f(f(x, \kappa_f(x))) + \lambda(f(x, \kappa_f(x))) - V_f(x) - \lambda(x) \\
&= V_f(x) - \ell(x, \kappa_f(x)) + \ell(x^e, u^e) + \lambda(f(x, \kappa_f(x))) - V_f(x) - \lambda(x) \\
&= -\ell(x, \kappa_f(x)) + \ell(x^e, u^e) + \lambda(f(x, \kappa_f(x))) - \lambda(x) \\
&= -[\ell(x, \kappa_f(x)) - \ell(x^e, u^e) + \lambda(x) - \lambda(f(x, \kappa_f(x)))] \\
&= -L(x, \kappa_f(x))
\end{aligned}$$

To establish asymptotic stability, we must verify that $\tilde{V}_f(x)$ serves as a Lyapunov function for the closed-loop system:

1. $\tilde{V}_f(x) \geq \alpha_1(\|x - x^e\|)$ for some $\alpha_1 \in \mathcal{K}_\infty$

   - This holds because strict dissipativity ensures that $L(x, u) \geq \rho(\|x - x_e\|)$ for some $\rho \in \mathcal{K}_\infty$
   - Since V_f is continuous with $V_f(x^e) = 0$, and $\lambda(x)$ satisfies the strict dissipativity condition, their sum $\tilde{V}_f$ is positive definite with respect to x_e

2. $\tilde{V}_f(x) \leq \alpha_2(\|x - x^e\|)$ for some $\alpha_2 \in \mathcal{K}_\infty$

   - This follows from the continuity of V_f and λ on the compact set X_f

- Both functions cancel at x^e, so their sum can be upper-bounded by an appropriate $\mathcal{K}_\infty$ function

3. $\tilde{V}_f(f(x, \kappa_N(x))) - \tilde{V}_f(x) \leq -\alpha_3(\|x - x^e\|)$ for some $\alpha_3 \in \mathcal{K}_\infty$

   -

$$\begin{aligned} &\tilde{V}_f(f(x, \kappa_N(x))) - \tilde{V}_f(x) \leq -L(x, \kappa_N(x)) \\ &\text{and} \qquad\qquad L(x, \kappa_N(x)) \geq \rho(|x - x_e|) \end{aligned}$$

Therefore we can take $\alpha_3 = \rho$ to satisfy this condition.

To complete the proof of asymptotic stability, we must establish attractivity that the state trajectory convergesto the equilibrium point x^e for $k \to \infty$.

Consider the sequence of states $\{x(k)\}_{k=0}^{\infty}$ generated by the closed-loop system $x(k+1) = f(x(k), \kappa_N(x))$ where κ_N is the first element of the optimal control sequence computed at time x(k). The initial state $x(0) \in \mathcal{X}$ The descent property of Lyapunov function becomes:

$$\tilde{V}_f(k+1) - \tilde{V}_f(k) \leq -\alpha_3(\|x(k) - x^e\|)$$

Since $\alpha_3(\|x(k) - x^e\|) \geq 0$ with equality if and only if $x(k) = x^e$. I have that $\{\tilde{V}_f(k)\}_{k=0}^{\infty}$ is a non-increasing sequence bounded below. Consequently, the sequence converges to some limit value $\tilde{V}_f^{\infty} \geq 0$.

Summing the inequality over all time steps, i have:

$$\sum_{k=0}^{\infty} [\tilde{V}_f(k+1) - \tilde{V}_f(k)] \leq -\sum_{k=0}^{\infty} \alpha_3(\|x(k) - x^e\|)$$

Developing to infinity and changing the sense of the inequality, I obtain:

$$\sum_{k=0}^{\infty} \alpha_3(\|x(k) - x^e\|) \leq \tilde{V}_f(0) - \tilde{V}_f^{\infty} < \infty$$

For this series to converge, we must have $\alpha_3(\|x(k) - x^e\|) \to 0$ when $k \to \infty$. Since $\alpha_3 \in \mathcal{K}_\infty$ this implies $\|x(k) - x^e\| \to 0$ for $k \to \infty$.

Thus, for any initial state within the feasible region $\mathcal{X}$, the state trajectory asymptotically converges to the equilibrium point x^e, establishing the attractivity property. Combined with the previously demonstrated Lyapunov stability, we conclusively establish that the equilibrium point x^e is asymptotically stable under the Economic MPC control law within the domain of attraction defined by the feasible region $\mathcal{X}$.

Chapter 5

Economic MPC Implementation: Design, Development, and Practical Considerations

After presenting the theoretical basis of Economic MPC and the description of the system, in this chapter, I introduce which cost functions and which constraints I used and how I implemented everything, using CasADi [1], a symbolic framework used to solve nonlinear optimization problems.

The implementation of the controller was done through Matlab, using 4 cost functions.

5.1 Cost Function

In selecting cost functions for this thesis, I aimed to develop formulations that would test different approaches to electric vehicle control. Starting from a function for simple tracking, I modified the cost function by adding economic parameters and removing tracking parameters which only remained inside the terminal cost. The 5 cost functions are the following:

5.1.1 Cost function 1: Reference Tracking

$$\begin{aligned} \ell_1(x_i, u_i) &= \|x_i - x_r\|_Q^2 + \|u_i\|_R^2 \\ V_{f,1}(x_N) &= \|x_N - x_r\|_P^2 \end{aligned} \tag{5.1}$$

where ℓ_1 represents a standard tracking cost function with quadratic penalty on state error and control effort. The matrix $Q = diag(0, \lambda_1, 0)$ weights state deviation from reference $x_r = [0, v_r, 0]^T$, R penalizes control input magnitude, and $P = diag(0, \gamma_1, 0)$ define the terminal state cost weighting. This approach focuses on following the reference trajectory as closely as possible while minimizing the control input value. The terminal cost $V_{f,1}(x_N)$ ensures that the final state approaches the reference I use this cost function, together with the CTG controller, as a basis to be able to make comparisons for cost functions with a more economical formulation.

5.1.2 Cost function 2: Power-Based Approach

$$\begin{aligned} \ell_2(x_i, u_i) &= \|x_i - x_r\|_Q^2 + \mu_2 \cdot \left\| \frac{P_{\text{em},i}}{P_{\text{ref}}} \right\|^2 \\ V_{f,2}(x_N) &= \|x_N - x_r\|_P^2 \end{aligned} \tag{5.2}$$

For the second cost function, I replaced the control effort penalty with a term that penalizes motor power consumption. The electric motor power was defined in chapter 3 and here $P_{em,i}$ represents the electric motor power at prediction step i, normalized with a reference value P_{ref}. This modification shifts focus toward energy efficiency while maintaining tracking performance through the state error term. The power term creates a more direct relationship with battery consumption, encouraging the controller to find trajectories that achieve reference tracking with reduced energy usage. The terminal cost is equal at cost function 1.

5.1.3 Cost function 3: Economic Battery Focus

$$\begin{aligned} \ell_3(x_i, u_i) &= \lambda_3 \cdot \dot{\zeta_i}^2 + \mu_3 \cdot \left\| \frac{P_{\text{em},i}}{P_{\text{ref}}} \right\|^2 \\ V_{f,3}(x_N) &= \|x_N - x_r\|_P^2 \end{aligned} \tag{5.3}$$

The third cost function represents my first true economic formulation. I removed all tracking terms from the stage cost, replacing them with terms that directly penalize battery discharge rate $\dot{\zeta}_i^2$ and normalized motor power. The term $\lambda_3 \cdot \dot{\zeta}_i^2$ penalizes rapid changes in the State of Charge (SOC), while the power term, as in the previously cost function, discourages excessive energy consumption. The terminal cost is the same as before, like in the cost function 1 and 2.

5.1.4 Cost function 4: CTG-Enhanced Economic Approach

$$\begin{aligned} \ell_4(x_i, u_i) &= \lambda_4 \cdot \dot{\zeta}_i^2 + \mu_4 \cdot \left(\frac{P_{\text{em},i}}{P_{\text{ref}}} \right)^2 + \gamma_4 \cdot (u_i - u_{\text{CTG},i})^2 \\ V_{f,4}(x_N) &= \|x_N - x_r\|_P^2 \end{aligned} \tag{5.4}$$

The fourth cost function is the same of the third cost function with additional Constant Time Gap (CTG) control term. This new term is obtained starting from the CTG policy illustrated on the paragraph 3.7. At each prediction step, the controller calculates a reference control input $u_{CTG,i}$ based on CTG logic, and penalizes deviations from this value. The CTG controller provides a reference input designed to maintain a safety distance that depend that adapts proportionally with the vehicle's speed. This choice of hybrid cost function attempts to balance economic benefits of energy conservation with the good following behavior inherent to CTG-based control.

5.1.5 Cost function 5: CTG Terminal Economic MPC

$$\begin{aligned} \ell_5(x_i, u_i) &= \lambda_5 \cdot \dot{\zeta}_i^2 + \mu_5 \cdot \left(\frac{P_{\text{em},i}}{P_{\text{ref}}} \right)^2 \\ V_{f,5}(x_N) &= \|x_N - x_{CTG}\|_M^2 \end{aligned} \tag{5.5}$$

where $x_{CTG} = [0, v_{CTG}, 0]^T$ and $M = diag(0, \gamma_5, 0)$. The final cost function uses the same economic stage cost of ℓ_3 but replace the terminal velocity reference with a velocity delivered from the CTG algorithm. This formulation allows energy optimization during transient phase while ensuring the terminal behavior conforms to the CTG velocity profile.

These cost functions represent a gradual transition from the simple tracking of a vehicle to the economic cost function, thus taking into account the transient and economic cost of tracking the previous vehicle.

5.2 Constraint formulation for the implementation

After describing all the cost functions I implemented, it is now necessary to present all the constraints placed within my problem, some of which have already been anticipated in the previous chapter, during the general formulation.

5.2.1 State Constraints

In order to limit the solutions of the minimization problem in the feasible set, I decided to impose the following bounds:

$$pos_{min} \leq x_1 \leq \quad pos_{max} \text{ for } i = 1, \dots, N \tag{5.6}$$

$$v_{min} \leq x_2 \leq \quad v_{max} \text{ for } i = 1, \dots, N \tag{5.7}$$

$$\zeta_{min} \leq x_2 \leq \quad \zeta_{max} \text{ for } i = 1, \dots, N \tag{5.8}$$

where :

- $x_{min} = 0\ m$ all my cycles are made to have a positive final displacement, my car cannot arrive at a negative point;
- $x_{max} = 25000\ m$ while running the CTG, I noticed that regardless of the reference signal input, my vehicle never moved more than 24000 meters, so I decided to set the upper limit to 24000 plus a safety value;
- $v_{min} = -5\ m/s$ this allows for small backward movements if needed;
- $v_{max} = 50\ m/s$ which corresponds to 180 km/h, the maximum speed tha car can reach;
- $\zeta_{min} = 0.01$ to avoid numerical problems when SOC approaches zero
- $\zeta_{max} = 1$ representing a fully charged battery

5.2.2 Input Constraints

The input constraints represent the physical limitations of the motor, with values derived from the efficiency map I analyzed in Chapter 3. These constraints ensure the optimization remains within the motor's operational capabilities:

$$T_{em,min} \leq u_i \leq T_{em,max} \text{ for } i = 0, \ldots, N-1 \tag{5.9}$$

Looking the efficiency map data, I determined that $T_{em,min} = -T_{em,max} = -280\, Nm$ represent torque limits for the electric motor in the Fiat 500e. During early testing phases, I noticed that unconstrained optimization sometimes suggest torque values that exceed these physical limits during acceleration scenarios. Such solutions, while mathematically optimal, would be physically unrealizable. Implementing these constraints ensures the controller generates commands that the actual vehicle can execute.

5.2.3 Vehicle Following Constraints

To ensure proper vehicle following behavior, I needed constraints that would prevent both dangerous tailgating and excessive delays. Safety demands maintaining sufficient distance from the lead vehicle, while practical following behavior requires staying within a reasonable working range of the sensor.

First, I implemented a minimum distance constraint to avoid potential collisions:

$$x_{1,i} - x_{ref} + min\_gap \leq 0 \text{ for } i = 1, \ldots, N \tag{5.10}$$

where $min\_gap = 0.5\, m$ meters provides a small safety buffer. This constraint prevents my vehicle from collision the lead vehicle.

Then I added a maximum distance constraint to ensure the vehicle remains within sensor range:

$$x_{ref} - x_{1,i} - max\_gap(x_{2,i}) \leq 0; \text{for}; i = 1, \ldots, N \tag{5.11}$$

For this constraint, I defined the maximum allowed gap as velocity-dependent:

$$max\_gap(x_{2,i}) = base\_gap + time\_factor \cdot x_{2,i} \tag{5.12}$$

where $base\_gap = 5\,m$ represent the maximum allowable distance at zero speed, and $time\_factor = 6\,s$ determines additional distance permitted per unit velocity.

This creates an adaptive following distance increasing with speed. I found that a time factor of 6 seconds offers good balance between energy efficiency and following behavior. Higher speeds with greater following distances allow more flexibility in energy management through smoother acceleration/deceleration profiles.

5.2.4 Terminal Constraint

For asymptotic stability, I added a terminal constraint that forces the final velocity at the end of prediction horizon with the reference velocity. This constraint is added with the following equality constraints:

$$x_{2,N} - v_{ref} = 0 \tag{5.13}$$

This constraint is equal for all the cost function.

The terminal constraint proved essential during the initial test phases. Without it, the controller exhibited energy efficient but impractical behavior, like stay in the starting position, or maintaining excessive distances that would eventually lead to complete separation from the lead vehicle.

5.2.5 Dynamics Constraints

The most important constraints are the dynamics constraints that ensure the system follows the vehicle model derived in Chapter 3:

$$x_{i+1} = f(x_i, u_i); \text{for}; i = 0, \ldots, N-1 \; x_0 \qquad = x_{current} \tag{5.14}$$

These constraints are implemented following the same equations, used to simulate a step forward of the system. But unlike them, it was necessary to use the native interpolation functions of casadi and not those of Matlab, since those of CasADi allowed the use of symbolic variables, while those native to Matlab did not. Another problem beyond this change of interpolator, arose for the interpolation of efficiency, after several attempts to use a multi-variable interpolation within CasADi, I did not find any acceptable solution. For this reason, I have, in agreement with Professor Pagone, kept the value of efficiency constant during the prediction horizon, since it is short-lived.

5.3 Implementation of CTG Controller

The Constant Time Gap (CTG) approach forms a key part of my control strategy, especially for cost functions 4 and 5. Unlike my EMPC implementation which does optimization over a prediction horizon, the CTG controller provides a direct control law based on current state. The CTG is implemented following the equation presented in the section 3.7

For equation 4, it is used in the form of the output torque required to reach the CTG speed, inside the stage cost. The acceleration required to reach the CTG speed has been converted into torque, also taking into account the resistive forces:

$$F_{required} = M_{vehicle} \cdot a_{command} + F_{resistive} \tag{5.15}$$

$$T_{wheel} = F_{required} \cdot r_{wheel} \tag{5.16}$$

$$u_{CTG} = \frac{T_{wheel}}{\tau_{gearbox} \cdot \eta_{gearbox}^{sign(T_{wheel})}} \tag{5.17}$$

where $a_{command}$ is the acceleration necessarily to maintain the desired time gap. The computation of the CTG torque is performed taking into account the predicted position and velocity at time i with respect to the reference position and velocity.

For cost function 5, instead, the CTG is included through the terminal cost. Here the equation of the necessary acceleration written inside the CTG section is converted into speed through the simple formula:

$$v_{CTG} = v_{current} + a_{command} \cdot T_s \tag{5.18}$$

where $v_{current}$ is the actual velocity of the vehicle and T_s is the sampling time of the system. In this case, in order to calculate v_{CTG} I consider the velocity and position at the prediction instant N.

5.4 Practical Implementation and problem in MATLAB

To implement my controller design, I developed a different Matlab files organized according to their functional roles. This modular approach facilitated development and debugging while maintaining separation between simulation environment and controller implementation.

5.4.1 Evolution of Implementation Strategy

My initial implementation strategy envisioned a hybrid approach: controller logic is implemented within a Matlab function while the plant dynamics would be simulated through Simulink environment. This approach seemed promising in the initial stage of my thesis because Simulink offering a visual environment to implement the dynamic of the system.

However, this plant encountered significant obstacles. The "MATLAB Function" block within Simulink, while theoretically suitable for embedding complex controller logic, proved remarkably inefficient when handling the computational demands of nonlinear optimization. Early tests revealed simulation times that were prohibitively long extending to hours even for relatively short driving cycles. This computational bottleneck primarily stemmed from Simulink handling of the CasADi symbolic framework within the MATLAB Function block, which created significant overhead with each function call.

After thorough testing and performance analysis, I opted to abandon the Simulink implementation in favor of a pure MATLAB approach. I faced a decision between two alternatives: either replacing the problematic "MATLAB Function" block with another Simulink block (which created its own complications with certain operations, particularly the interpolation functions needed for battery voltage and resistance calculations), or migrating the entire implementation to MATLAB script files. The latter option proved substantially more efficient, decreasing simulation times significantly while providing better control over memory management, more straightforward integration with the CasADi framework, and enhanced capabilities for troubleshooting the optimization solver's behavior during execution.

5.4.2 System Architecture

The implementation consists of two primary components:

- A main simulation file that orchestrates the entire process.

- A controller function file that encapsulates the optimization problem.

The main file simulates vehicle behavior through a time-stepping approach, iterating from $t = 0$ to $t = T_{fin}$ (representing the total duration of the driving cycle). At each time step $T_s = 0.5\,s$ the controller function is called to formulate and solve the optimization problem based on current conditions. The controller returns the optimal control input, which is then applied to advance the system state. This updated state becomes the initial condition for the subsequent optimization problem.

5.4.3 Numerical Challenges in Battery Model Implementation

Another significant challenge I encountered emerged when implementing the equation for battery current calculation (3.17) within the MATLAB environment. During several simulation runs, CasADi repeatedly failed to converge to a solution for the optimization problem, generating error messages that initially appeared cryptic. After meticulously stepping through the code during debugging sessions, I identified the root cause: mathematical instability occurring when the discriminant in the quadratic formula approached zero, causing the entire optimization to collapse.

The solution required a subtle but effective numerical workaround that preserved the mathematical integrity of the model:

```
discriminant = VB^2 - 4 * RB * PBattery;
discriminant_safe = max(discriminant, 0.01);
IB_k = (VB - sqrt(discriminant_safe)) / (2 * RB);
SOC_dot = -(IB_k / (params.Q_nom * 3600 * params.eta_c^sign(IB_k)));
```

This small modification establishing a minimal threshold for the discriminant proved remarkably effective at preventing numerical singularities without significantly affecting the physical accuracy of the model.

This experience highlighted a crucial aspect of implementing controllers inside computers: theoretical mathematical formulations often require subtle adjustments to function robustly in computational environments. The idealized assumptions of continuous mathematics must sometimes be reconciled with the discrete, finite precision reality of numerical computing systems.

5.4.4 Selection of Prediction Horizon

During my implementation, choosing a good prediction horizon (N) was a key factor for balancing how fast the calculations would run and how well the controller would work. I did some tests with different horizon lengths to see their effects.

The results in Table 5.1 shows clearly what happens when i change the prediction horizon across different cost functions.

N	J_1		J_2		J_3		J_4		J_5		CTG
	SOC	Time[s]	SOC	Time[s]	SOC	Time[s]	SOC	Time[s]	SOC	Time[s]	
2	0.6926	93	0.6926	93	0.6926	94	0.6926	112	0.6926	102	0.6934
3	0.6934	120	0.6934	117	0.6924	114	0.6926	193	0.6934	131	↓
4	0.6931	133	0.6930	138	0.6932	136	0.6845	235	0.6945	142	↓
5	**0.6938**	**152**	**0.6932**	**156**	**0.6938**	**154**	**0.6523**	**265**	**0.6951**	**161**	↓
6	0.6941	174	0.6936	175	0.6940	173	0.6919	241	0.6953	185	↓
7	0.6949	186	0.6940	196	0.6948	195	0.6942	341	Error	Error	0.6934

Table 5.1: Comparison of final SOC values and computation times for different cost functions and prediction horizons using the WLTP driving cycle

As expected, computation time gets longer with bigger prediction horizon. For example, with cost function J_1, the computation time increases from about 93 seconds with N=2 to around 186 seconds with N=7. This makes sense because larger N values mean the optimizer needs to solve a bigger problem with more variables.

Looking at the SOC values, I notice some interesting patterns. For most cost function, the final SOC value improves slightly as the prediction horizon gets longer. This shows that when the controller can "see" further ahead, it makes better decisions about energy use.

Cost function J_4 shows the most dramatic impact from prediction horizon changes. With N=5, it achieves the lowest SOC value of 0.6523, which is much lower than any value achieved by MPC tracking or CTG.

For cost function J_5, I encountered numerical issues when trying to use N=7, resulting in solver errors. This highlights that very long prediction horizons can sometimes create convergence problems.

After analyzing these results, I decided to use N=5 for my main experiments. This value offers a good compromise between:

- Computation time (still reasonable at 151-265 seconds depending on cost function)
- Control performance (SOC values generally better than shorter horizons)
- Numerical stability (no convergence issues observed)

Also, with N=5 and sampling time $T_s = 0.5\,s$, the controller looks ahead 2.5 seconds, which matches well with typical driver reaction times in traffic following scenarios.

5.4.5 Solver Configuration

The final implementation aspect involved configuring the optimization solver. I used IPOPT through the CasADi interface with the following key settings:

```
opts = struct;
opts.ipopt.max_iter = 300;
opts.ipopt.tol = 1e-3;
```

I set the maximum iteration count to 300 proved sufficient convergence in most cases, while the tolerance of $1 \cdot 10^{-3}$ balanced solution accuracy against computation time. I experimented with tighter tolerances $(10^{-4}, 10^{-5})$ but found minimal improvement in control performance despite significantly increased computation times. Similarly, reducing max iterations below 300 occasionally led to premature termination without finding optimal solutions.

5.5 Simulation Testing Methodology

In order to evaluate the performance of the various controllers, I designed a progressive testing methodology. Before attempting to analyze the controllers with complex real-world driving profiles, I needed to verify their basic functionality with simpler reference trajectories.

As previously mentioned, I tested my controllers through progressively complex scenarios. This approach let me debug basic functionality before tackling realistic driving conditions.

- **Step 1 - Constant Velocity reference**: I first tested the controller with a simple constant velocity of $11.11\,m/s$ (approximately $40\,km/h$). This initial test checked steady-state stability and basic tracking ability of my implementation. During these tests I observed how each cost function approached the reference speed and maintained it over time.

- **Step 2 - Sinusoidal velocity reference**:Next I introduced predictable but continuous changes to test controller response to transitions. This pattern helped me understand how each formulation handles periodic accelerations and decelerations.

- **Step 3 - WLTP cycle testing**: Only after confirming proper operation with simpler tests did I move to the standardized WLTP driving cycle. This cycle represents realistic driving with varied accelerations, steady speeds and multiple deceleration phases. During the first two implementation phases, I mainly adjusted cost function parameters to verify their impact on reference tracking behavior. In this final step, I performed a general analysis to optimize parameters for the final cost functions based on energy efficiency metrics.

This stepwise approach proved critical for my implementation. Instead of directly testing with the complex WLTP cycle, I used this gradual method to fix issues at each level. Starting with the WLTP cycle immediately would have created overwhelming debugging problems, making it impossible to know if errors came from basic controller design, constraints, or cycle features. Working through simpler tests first saved time and reduced frustration, especially when tuning the economic cost functions that reacted strongly to changes in prediction horizon and constraints. This approach let me solve basic issues with simple references first, avoiding many hours of debugging in the more complex scenarios.

Chapter 6

Simulation Results, Parameter Selection, and Conclusions

After presenting the theoretical framework and MPC setup with their respective cost functions in previous chapters, I now focus on simulation results from all three testing phases. This chapter explains my parameter value selection process and analyzes performance differences between control strategies

6.1 Phase 1: Constant Velocity Reference

As mentioned in the last section of the previous chapter, the first testing phase evaluated controller performance with a constant reference velocity of $11.11\,m/s$ $(40\,km/h)$. This simplified scenario allowed me to observe the basic characteristics of each control approach, particularly how they converge to steady-state velocity values.

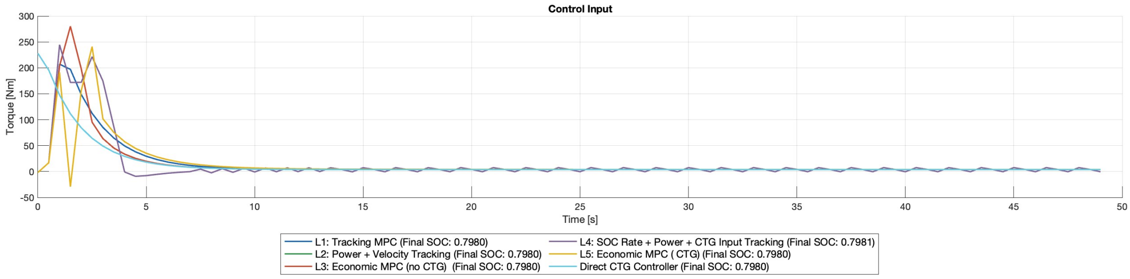

Figure 6.1: Control input comparison for different controllers during constant velocity reference test

6.1.1 Control Input Analysis

The Figure 6.1 shows the torque commands generated by different controllers during the first 50 seconds of simulation. All controllers initially generate high torque to reach the target speed. Only at a single instant of time do some of the cost functions reach 280 Nm, the physical limit of torque delivered by the motor. In the remaining time, they remain within the acceptable torque range.

In steady state, all controllers maintain similar small oscillations in torque commands. These oscillations remain minimal for most cost functions, with J_4 exhibiting slightly larger variations (about $\pm 5\,Nm$). Despite these minor fluctuations, passenger comfort and energy consumption are not significantly affected.

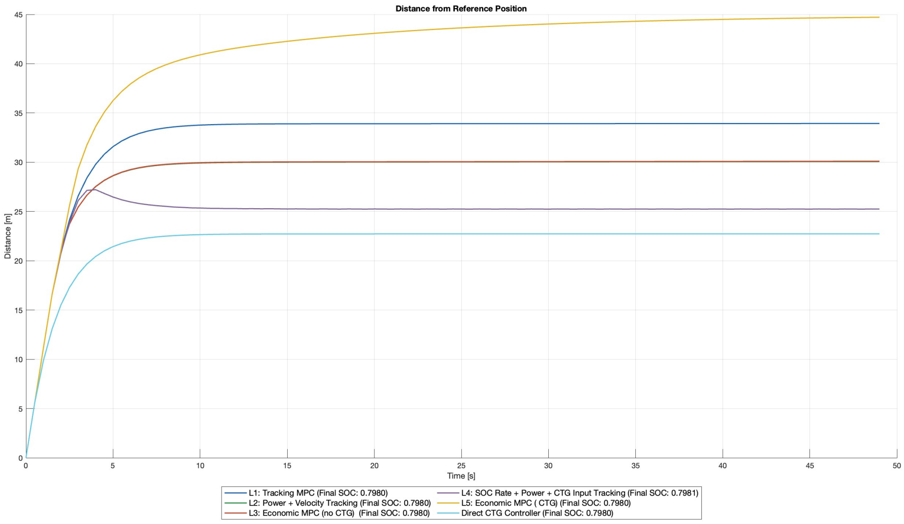

Figure 6.2: Distance from reference position for different control strategies

6.1.2 Distance Management Strategies

Figure 6.2 shows differences in following distance strategies. The CTG controller maintains the smallest gap ($22\,m$), while the cost function J_5 (Economic MPC with CTG terminal constraint) allows the larges separation, approaching 45 meters at steady state. The remaining controllers settle at intermediate distances between these extremes.

These steady-state distances stem directly from each controller's optimization priorities. J_5 maximizes energy conservation by maintaining a larger gap, allowing more flexibility to optimize acceleration profiles. The CTG controller prioritizes close following without considering energy implications. The differences reflect fundamental trade-offs between efficient vehicle operation and tight following behavior.

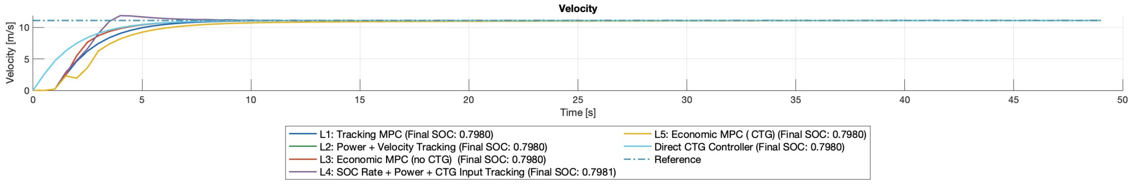

Figure 6.3: Velocity evolution during constant velocity test

6.1.3 Velocity Tracking performance

Figure 6.3 displays the velocity profile of all controllers, which as can be seen reach the target velocity of $11.11\,m/s$. The main differences appear in the transient approach. J_4 (SOC velocity + power + CTG input tracking) shows an overshoot. The second cost function J_2 (power + velocity tracking) reaches the target velocity the fastest, while J_5 (Economic MPC with CTG) shows the slowest approach, taking about $20\,s$ to reach the steady state.

6.2 Phase 2: Sinusoidal Velocity Reference

After validating the controllers with constant velocity, I moved to test their response to a sinusoidal velocity profile that oscillated between $40\,km/h\,(11.11\,m/s)$ and $60\,km/h\,(16.67\,m/s)$. This pattern creates a more demanding scenario that better reveals the differences between control strategies when handling the acceleration and deceleration phases.

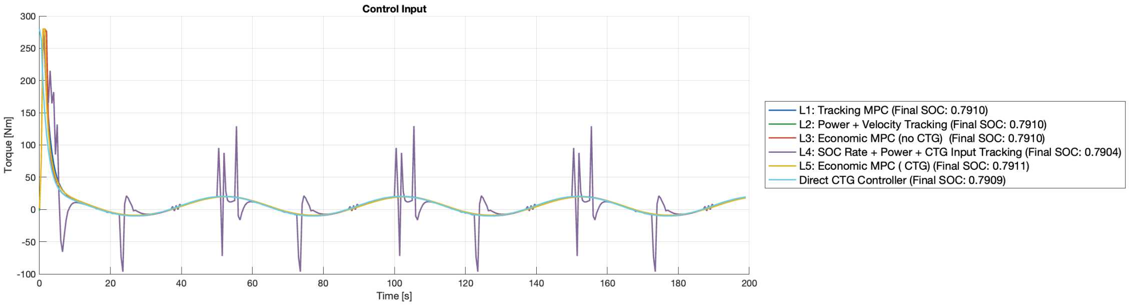

Figure 6.4: Control input comparison for different controllers during sinusoidal velocity reference test

6.2.1 Control Input Analysis

Figure 6.4 shows torque commands from the different cost functions during the sinusoidal reference test. Unlike the constant velocity test, this test reveals more distinct behaviors between controllers.

From the graph we can see how all cost functions reach an initial peak of 280 Nm to approach the reference velocity. Only J_4 shows a spiking behavior, reaching about 250 Nm at several points during the simulation. Throughout the entire simulation, it exhibits a pattern with both positive and negative peaks. I've attributed this behavior to computational artifacts, since when I modified parameters within the cost function , these spikes would shift position, reduce in amplitude, or disappear entirely.

The other cost functions show similar behavior to each other. They typically have a single peak at 280 Nm and then maintain smoother torque profiles that closely resemble each other. This suggests that despite their different formulations, most controllers converge to similar steady-state control strategies when following a sinusoidal pattern.

6.2.2 Distance Management Performance

The distance plot in Figure 6.5 reveals significant differences in following behavior. Each controller maintains a consistent pattern relative to the reference vehicle, but with varying average distances and amplitude variations.

J_5 (Economic MPC with CTG terminal constraint) maintains the largest average gap, with distances oscillating between $45 - 62$ meters. This large separation provides maximum flexibility for energy optimization, allowing the vehicle to adapt its speed profile with minimal torque adjustments.

The other cost functions maintain intermediate distances between the J_5 and the CTG controller, with cost function J_4 showing behavior closer to the CTG with slightly larger distances. The consistent oscillation patterns indicate that all

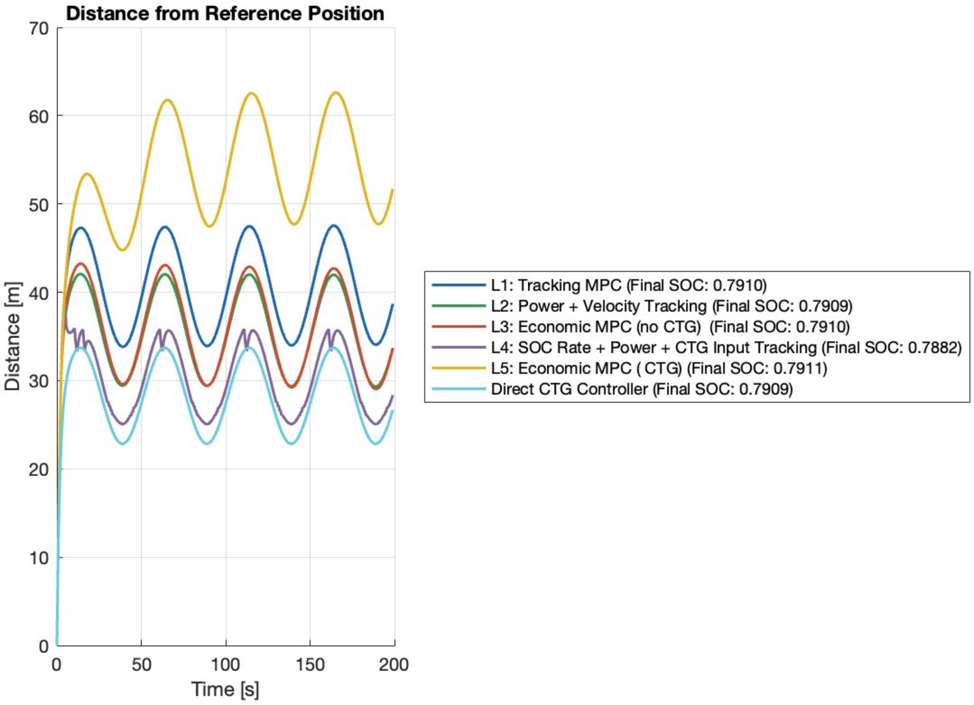

Figure 6.5: Distance from reference position for different control strategies

controllers have found stable following behaviors that match their optimization priorities.

An interesting observation is how the distances change during acceleration vs. deceleration phases. During acceleration, the gaps typically increase as controllers balance energy use against closing the distance. During deceleration, the controllers allow the gaps to naturally reduce.

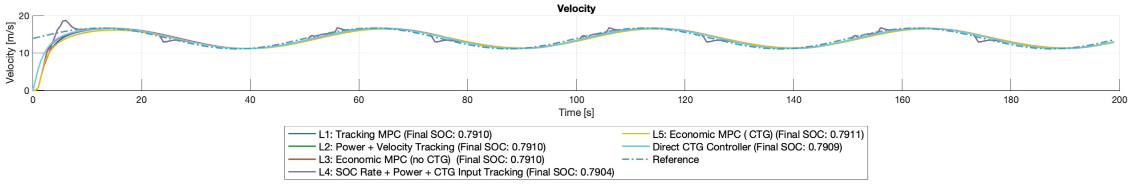

Figure 6.6: Velocity evolution during sinusoidal velocity test

6.2.3 Velocity Tracking Performance

The velocity profiles in Figure 6.6 show that all controllers tracking the sinusoidal velocity profile, but with different tracking characteristics.

J_4 exhibits several speed peaks, consistent with its aggressive torque approach. These overshoots reach about 2 m/s above the reference maximum speed reached by the reference, which can potentially causing passenger discomfort. This behavior aligns with the torque spikes observed earlier, suggesting that this controller prioritizes gap control at the expense of smooth velocity transitions.

The J_5 function never quite reaches the maximum reference speed, peaking at approximately 16,1 m/s compared to the reference 16.67 m/s. On the descending portions of the cycle, it maintains a smoother profile by consistently keeping a higher speed, which helps maintain an acceptable distance from the reference vehicle. This behavior reflects its economic optimization priority, sacrificing exact tracking to achieve smoother, more efficient operation.

The other cost functions follow the reference in remarkably similar ways, showing behaviors that are slightly out of phase with the reference. This phase difference represents the natural delay in response as the controllers balance tracking performance against their other objectives.

6.2.4 Initial Observation on Energy Efficiency

While a detailed energy analysis will be covered in the WLTP cycle tests, some preliminary observations can be made from the sinusoidal tests.

The approach used by cost function J_4 is aggressive and requires more energy due to its abrupt torque commands and velocity overshoots. On the other hand, cost function J_5 appears more energy-conscious, maintaining greater following distances and smoother speed profiles which should translate to lower energy consumption. This more conservative approach avoids unnecessary acceleration and deceleration cycles, potentially preserving battery charge through more gradual power demands.

These initial findings indicate a clear trade-off between tight following behavior and energy optimization. Controllers that maintain larger following distances generally exhibit smoother operation and likely better energy efficiency, while those prioritizing close following behavior show more aggressive control inputs.

6.3 Parameter Tuning and Optimization Analysis

Before presenting the WLTP cycle results, I want to discuss the parameter tuning process I conducted to minimize battery consumption while satisfying the constraints.

The process of finding the best parameters for my cost functions required numerous simulations with WLTP cycle, to see what changed. I started from a neutral cost function with all parameters set to 1, then increased one parameter at a time until I reached the final cost function values.

6.3.1 Parameter Analysis for Cost Function 1

Recalling that cost function J_1 is defined as:

$$J_1 = \sum_{i=0}^{N-1} \alpha_1 (v_i - v_{ref})^2 + \beta_1 u_i^2 \quad + \gamma_1 (v_N - v_{ref})^2$$

α_1	β_1	γ_1	SOC
1	1	1	0.6935
1	5	1	0.6935
5	1	1	0.6935
1	1	5	0.6935
1	5	5	0.6935
5	1	5	0.6935
10	1	10	0.6935
1	**10**	**10**	**0.6936**

Table 6.1: Parameter values and corresponding SOC for cost function J_1

Table 6.1 shows my testing results for various parameter combinations in cost function J_1. During this analysis, I observed minimal variation in final SOC values despite changes in weighting parameters. Most configurations yielded identical SOC values of 0.6935, with only marginal improvement to 0.6936 in the final selected configuration. When increasing α_1 while keeping other parameters constant, the controller becomes more aggressive in tracking velocity, but this aggressiveness in reaching the tracking velocity doesn't translate to a change in SOC. Modifying γ_1 also doesn't lead to changes regarding the SOC. Meanwhile, modifying parameter β_1 led to a minimal change in SOC. The higher torque penalty likely encourages smoother control actions, reducing unnecessary acceleration and deceleration cycles.

6.3.2 Parameter Analysis for Cost Function 2

The cost function J_2 is defined as:

$$J_2 = \sum_{i=0}^{N-1} \alpha_2 (v_i - v_{ref})^2 + \beta_2 \left(\frac{P_{em}}{P_{ref}} \right)^2 \quad + \gamma_2 (v_N - v_{ref})^2$$

Table 6.2 shows how different parameters affect the final SOC when using cost function J_2. This function replaces the torque penalty from J_1 with a term that directly penalizes motor power.

α_2	β_2	γ_2	SOC
1	1	1	0.6935
1	5	1	**0.6936**
5	1	1	0.6934
1	1	5	0.6935
1	5	5	**0.6936**
5	1	5	0.6934
1	10	10	**0.6936**
10	1	10	0.6932

Table 6.2: Parameter values and corresponding SOC for cost function J_2

The parameter β_2 directly weights the power term. When increasing this value from 1 to 5, I observed a small improvement in final SOC. This makes sense because higher values of β_2 more aggressively penalize power usage. I also noticed that increasing α_2 (the velocity tracking weight) actually worsened the final SOC slightly. With $\alpha_2 = 10$, the final SOC dropped to 0.6932, showing that aggressive velocity tracking can reduce energy efficiency. This happens because the controller forces the vehicle to match reference speed exactly, even when a small deviation might save energy. The terminal weight γ_2 showed minimal impact on final SOC, similar to what I observed with J_1. Since my goal is to minimize the energy consumed to follow the reference, one of the cost functions that minimizes the SOC consumption is fine.

6.3.3 Parameter Analysis for Cost Function 3

The cost function J_3 represents my transition to a true economic formulation. Unlike the previous cost functions, it removes direct velocity tracking from the stage cost and focuses on energy-related terms:

$$J_3 = \sum_{i=0}^{N-1} \alpha_3 \dot{\zeta}^2 + \beta_3 \left(\frac{P_{em}}{P_{ref}} \right)^2 \quad + \gamma_3 (v_N - v_{ref})^2$$

Table 6.3 shows the impact of different parameter combinations on final battery state.

α_3	β_3	γ_3	SOC
1	1	1	0.6924
5	1	1	0.6923
1	5	1	**0.6937**
1	1	5	0.6922
5	1	5	0.6924
1	5	5	**0.6937**
1	10	10	**0.6937**
10	1	10	0.6923

Table 6.3: Parameter values and corresponding SOC for cost function J_3

With this cost function, the parameter α_3 weights the squared rate of change of SOC ($\dot{\zeta}^2$). Increasing this value from 1 to 5 actually decreased SOC slightly, from 0.6924 to 0.6923. This was unexpected since I thought penalizing SOC change would improve efficiency. After analyzing the results, I realized that this might be due to the MPC's receding horizon nature where only the first control input is applied. The optimization balances the entire predicted trajectory, but since we implement just the first step before recalculating at the next time instant, the controller thought to consume more in the initial states, and then consume less in the subsequent instants.

Parameter β_3 had the most significant impact. Increasing this value from 1 to 5 improved final SOC from 0.6924 to 0.6937, a larger improvement than seen in previous cost functions. This makes sense as β_3 directly penalizes motor power consumption. The controller becomes more reluctant to use power, finding trajectories that maintain adequate following while minimizing energy use.

The terminal cost weight γ_3 showed an unexpected effect. When increased from 1 to 5, the SOC decreased from 0.6924 to 0.6922. This suggests that enforcing stricter terminal tracking might sometimes require more energy near the end of the prediction horizon.

Overall, J_3 showed better energy conservation than both J_1 and J_2, confirming that an economic-focused formulation outperforms traditional tracking approaches for this application.

6.3.4 Parameter Analysis for Cost Function 4

The cost function J_4 represents a hybrid approach that combines economic objectives with the CTG control strategy. The cost function is:

$$J_4 = \sum_{i=0}^{N-1} \alpha_4 \dot{\zeta}^2 + \beta_4 \left(\frac{P_{em}}{P_{ref}} \right)^2 + \gamma_4 (u_i - u_{CTG,i})^2 \quad + \eta_4 (v_N - v_{CTG})^2$$

α_4	β_4	γ_4	η_4	SOC
1	1	1	1	0.6846
5	1	1	1	0.6644
1	5	1	1	0.6838
1	1	5	1	0.6547
1	1	1	5	0.6876
5	1	1	5	0.6645
1	5	1	5	0.6861
1	1	5	5	0.6548
10	1	10	20	0.6523
1	**10**	**1**	**10**	**0.6918**

Table 6.4: Parameter values and corresponding SOC for cost function J_4

The table 6.4 showed that the function J_4 have the most dramatic impact on energy consumption among all tested approaches. With appropriate parameter tuning, i have reach the final SOC value of 0.6918. Which is still lower than all the other functions and the CTG.

The parameter γ_4, which weights the deviation from CTG control input, proved most influential. Increasing the parameter from 1 to 5 involves a lowering of the SOC from 0.6846 to 0.6547, a significant loss. Interestingly, increasing α_4 (SOC rate penalty) from 1 to 5 reduced SOC from 0.6846 to 0.6644, showing that aggressive penalization of SOC changes doesn't always improve efficiency. Similarly, increasing β_4 (power penalty) from 1 to 5 slightly worsened SOC to 0.6838, contrary to what I observed in cost function J_3. This unexpected behavior illustrates the complex interactions between energy-focused terms and CTG-following objectives. The terminal constraint weight η_4 showed a positive impact. Increasing it from 1 to 5 improved SOC from 0.6846 to 0.6876, indicating that driving the system toward CTG-derived terminal conditions benefits overall energy economy. The best performance came from the configuration with $\alpha_4 = 1$, $\beta_4 = 10$, $\gamma_4 = 1$, and $\eta_4 = 10$. This cost function is the worst in terms of SOC performance. It's worth noting that this cost function was also the most sensitive to parameter changes, with SOC ranging from 0.6523 to 0.6918 across different configurations. This sensitivity indicates both the potential and the challenge of hybrid approaches that attempt to balance multiple control objectives.

6.3.5 Parameter Analysis for Cost Function 5

The cost function J_5 represents the culmination of my economic MPC approach, using a formulation that blends economic considerations with CTG-based terminal constraints:

$$J_5 = \sum_{i=0}^{N-1} \alpha_5 \dot{\zeta}^2 + \beta_5 \left(\frac{P_{em}}{P_{ref}} \right)^2 \quad + \gamma_5 (v_N - v_{CTG})^2$$

This function maintains economic terms in the stage cost while incorporating CTG concepts only in the terminal constraint.

α_5	β_5	γ_5	SOC
1	1	1	0.6944
5	1	1	0.6940
1	5	5	0.6938
1	1	5	0.6942
5	1	5	0.6942
1	5	5	0.6940
1	10	10	0.6939
1	**1**	**20**	**0.6951**

Table 6.5: Parameter values and corresponding SOC for cost function J_5

Table 6.5 shows the impact of different parameter combinations on the final SOC when using cost function J_5. Even with the initial version where all parameters are set to 1, this cost function shows better performance than any other I tested. The parameter α_5, which penalizes SOC rate changes, when increased from 1 to 5 leads to a decrease of 0.0004 in the final SOC. This reduction shows similar behavior to what I observed in cost function J_3. I notice different behavior with β_5, where increasing it from 1 to 10 doesn't bring improvement as it did in cost function J_3. This behavior suggests that increasing the stage cost weights causes me to lose emphasis on the terminal cost, which is the term that helps me conserve SOC. The terminal cost weight γ_5 shows the most influential impact on final SOC, as I expected. When increased from 1 to 5, the SOC slightly worsened to 0.6942. But when increased further to 20, it achieved the best performance with SOC $= 0.6951$. This value represents the highest SOC among all my tested controllers, indicating superior energy efficiency. What's particularly noteworthy about cost function J_5 is that it achieves excellent energy performance while maintaining good following behavior.

6.4 Phase 3: WLTP Cycle Test

After completing simulations on constant and sinusoidal signals and selecting the parameters for all cost functions based on final SOC and constraint satisfaction, I now present the results from the WLTP cycle test. This test is the most comprehensive evaluation as it divides into three phases representing typical car usage scenarios. With various speeds, accelerations, and decelerations, it allows assessment of controllers in a more practical setting.

Comparison of Control Approaches

The WLTP cycle tests revealed substantial differences in controller performance under realistic driving conditions. When examining the final SOC values across different approaches, cost function J_5 demonstrated the best energy efficiency, achieving a final SOC of 0.6951. This represents a measurable improvement over the direct CTG controller's 0.6934.

While the difference might appear modest, it translates to meaningful energy savings over longer trips. For the Fiat 500e with its 60 Ah battery, this improvement represents approximately 0.102 Ah saved during just one driving cycle of 1800s. Such efficiency gains become increasingly significant during daily use or longer journeys.

The most surprising observation from these tests was the poor performance of the J_4 cost function, which produced a final SOC of only 0.6918. This unexpected result suggests that attempting to directly track the CTG controller inputs within an economic framework creates counterproductive behavior, even worse than the classical CTG.

Conventional tracking such as cost function 1 and cost function 2, compared to CTG, show minimal improvements, with final SOC values, for both cost functions, of 0.6936. This is 0.02% higher than CTG. This mid-level performance confirms that, although traditional MPC formulations work adequately, they do not reach the efficiency potential that economic approaches can offer.

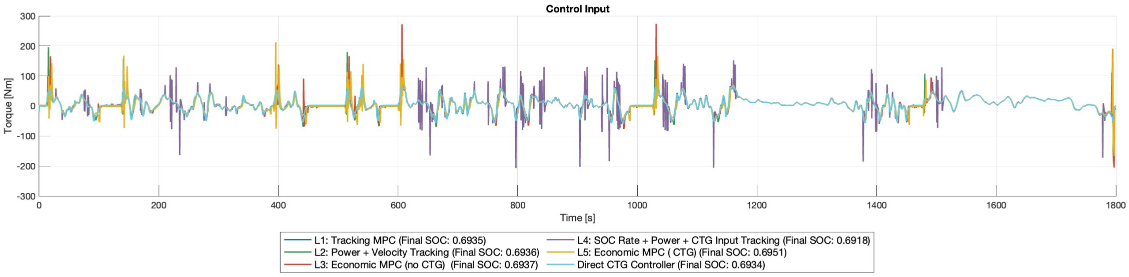

Figure 6.7: Control input comparison for different controllers during WLTP cycle test

6.4.1 Control Input Analysis

Analyzing control inputs in the Figure 6.7 across different strategies during the WLTP cycle, I noticed that this realistic driving scenario rarely pushes most cost functions to maximum torque limits.

The J_4 cost function exhibits noticeably erratic behavior, with torque commands fluctuating between $200\,Nm$ and $-200\,Nm$. These abrupt transitions between positive and negative torque values typically increase energy consumption through mechanical inefficiencies. This explains why J_4's final SOC value is the worst among all tested cost functions.

J_5 demonstrates the most effective control strategy from an energy perspective. It achieves a good balance between following behavior and energy optimization. Zooming into the graph reveals that during deceleration phases, J_5 maintains a smoother braking profile, extending the braking duration to recover more energy. This nuanced approach to regenerative braking contributes significantly to its superior energy conservation.

The traditional tracking approaches (J_1 and J_2) display similar characteristics to each other, prioritizing reference following without much consideration for battery consumption. These functions perform similarly to the standard CTG controller. Their nearly identical final SOC values reflect this shared approach despite differences in their mathematical formulations.

Analysis of these temporal separation profiles provides additional insight into J_5's superior energy efficiency. By permitting more generous time buffers, this controller implements smoother acceleration and deceleration profiles, thereby reducing energy-intensive torque fluctuations. The economic focus of J_5 prioritizes energy conservation within safety-acceptable temporal margins, whereas the CTG controller rigidly maintains shorter time separations regardless of energy implications.

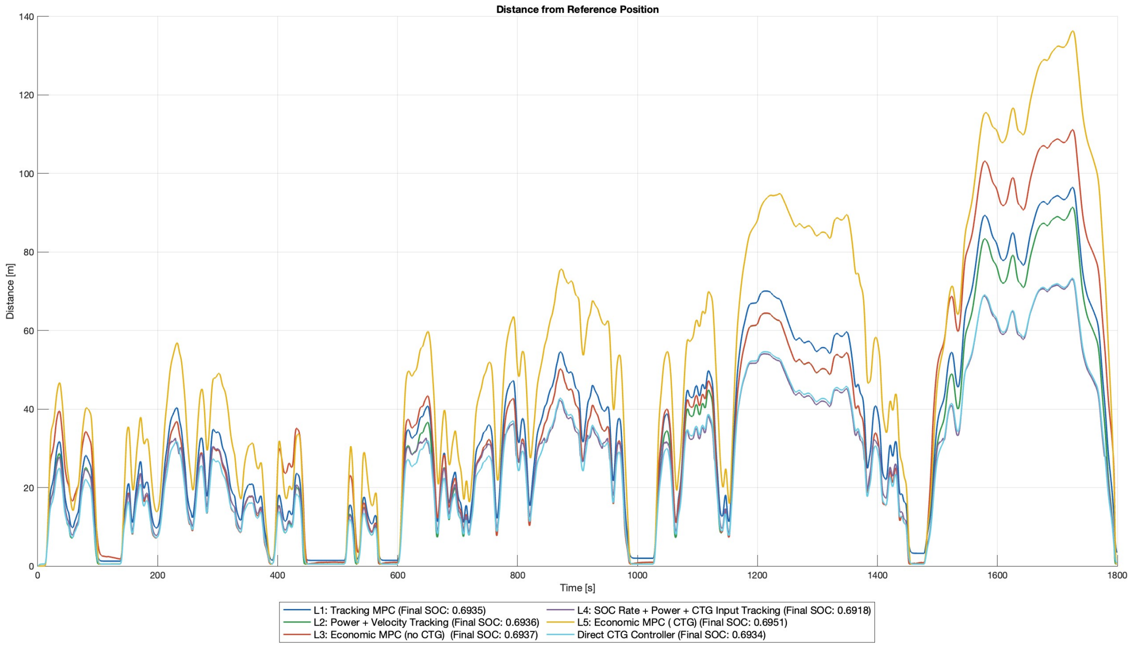

Figure 6.8: Distance from reference position for different control strategies

6.4.2 Distance Management Performance

The distance graph 6.8 in the WLTP cycle reveals significant differences, as already seen in the graphs of the previous phases, regarding the tracking strategies. Each controller has characteristic distance profiles, which were already visible in phase 2.

Cost function J_5 consistently maintains the largest following distance throughout the entire cycle, reaching a peak of $136\,m$ during high-speed segments when vehicle travel at approximately $130\,km/h$. this greater gap, as previously mentioned, allows for better management of transitions. On the other end, the CTG controllers maintains the smallest distance, reaching a maximum of only $75\,m$. In fact, the CTG prioritizes maintaining a standard time gap from the preceding vehicle without considering energy consumption.

Cost function J_4 exhibits the least clean distance profile. Zooming into the graph reveals that instead of maintaining smoothly increasing or decreasing distances, J_4 produces sharp peaks corresponding to torque spikes, causing distance variations of up to 2 meters within a single sampling period. This unstable behavior contributes to its poor energy performance, as the controller makes frequent corrections rather than maintaining smooth, energy-efficient operation.

The traditional tracking approaches (J_1 and J_2) show similar distance patterns to each other, maintaining intermediate gaps generally falling between the extremes of CTG and J_5. Their distance profiles demonstrate moderate responses to speed changes, maintaining gaps of $88 - 96$ meters during high-speed segments.

Another interesting section of the graph shows vehicle stopping and restarting behaviors. During these portions, CTG and J_5 reach the minimum distance of $0.5\,m$ while stopped, while other cost functions prefer maintaining greater distances, up to a maximum of $1.5\,m$ for J_1. When vehicles restart moving, the gap increases more rapidly with function J_5, indicating that this cost function doesn't simply try to catch up with the preceding vehicle but also considers energy costs in its calculations.

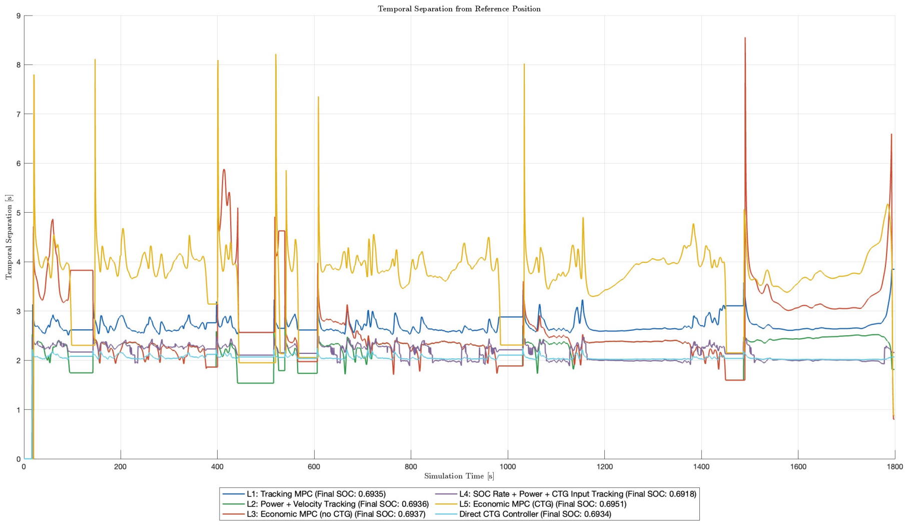

Figure 6.9: Temporal separation from reference position for different control strategies during WLTP cycle test. This parameter indicates how many seconds it would take for the following vehicle to reach the lead vehicle's position at current velocity.

6.4.3 Temporal Separation Analysis

Temporal Separation Analysis The distance graph provides information regarding the spatial separation between following and lead vehicles. However, temporal separation constitutes another critical parameter in vehicle-following dynamics. Figure 6.9 illustrates this parameter for all controllers throughout the WLTP cycle. This metric quantifies how many seconds it would take the following vehicle to reach the lead vehicle's position if continuing at its current velocity.

Examining temporal separation profiles reveals substantial differences between control strategies. The CTG controller maintains the most consistent temporal separation, approximately 2 seconds, which aligns precisely with its design parameters. This behavior emerges naturally from the CTG algorithm's fundamental objective of maintaining constant time gaps rather than fixed distances.

Cost function J_5 exhibits the greatest temporal separation throughout the cycle, with values averaging around 4.5 seconds, ranging from minimums of 2 seconds to maximums approaching 8 seconds. During high-speed segments, J_5 deliberately allows this temporal gap to increase slightly, reaching approximately 5 seconds. This behavior corroborates observations made in the spatial distance analysis. The extended time buffer enables J_5 to approach velocity transitions more gradually, which explains its superior energy efficiency performance.

J_4 exhibits an unstable temporal separation with abrupt changes that correspond to its erratic torque commands. These peaks in temporal separation sometimes reach 8 seconds before quickly falling back to 3 seconds, showing a lack of smooth control that leads to its poor power performance.

The traditional tracking controllers (J_1 and J_2) maintain intermediate temporal gaps, usually between 2.5-3 seconds. They show similar patterns to each other, which matches their similar SOC performance.

I notice also that all controllers show increased variations in temporal separation during low-speed phases of WLTP cycle. This happens because at low speeds, small changes in distance create bigger changes in time gap. When velocity term appears in denominator, calculation becomes more sensitive at low speeds.

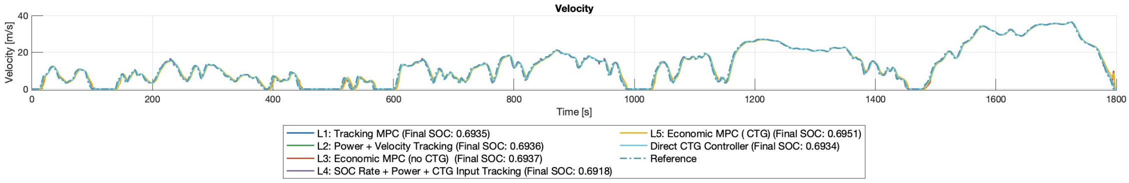

Figure 6.10: Velocity tracking performance across the entire WLTP cycle for all control strategies

6.4.4 Velocity Tracking Performance

Examining the velocity profiles in the Figure 6.10, we can observe that all controllers follow the reference velocity remarkably well throughout the entire cycle. This strong tracking performance occurs despite their different formulations and optimization priorities. The terminal velocity constraint likely plays a key role here, ensuring all controllers meet the reference velocity at the end of each prediction horizon regardless of their stage cost formulations.

Zooming into the graph reveals better differences in how controllers handle velocity. During acceleration phases, J_4 tends to have a more aggressive acceleration profile, with variations that explain the peaks in torque. On the other hand,J_5 shows a more moderate approach, managing accelerations and decelerations better, which explains its superior energy usage.

6.5 Conclusion

This thesis has explored the application of Economic Model Predictive Control to electric vehicle energy optimization during Adaptive Cruise Control scenarios. Through methodical testing across progressively complex driving conditions, I've identified meaningful differences between traditional tracking approaches and economic formulations.

My results demonstrate that Economic MPC with CTG terminal constraints (J_5) achieves measurably better energy efficiency while maintaining good following behavior. The final SOC value of 0.6951 represents a noticeable improvement over both traditional tracking approaches (0.6936) and the direct CTG implementation (0.6934). This efficiency gain comes mainly from how J_5 manages vehicle following distances and creates smoother acceleration and deceleration profiles.

Surprisingly, trying to directly incorporate CTG control inputs within the cost function (J_4) produced unexpectedly poor results. Despite seeming theoretically sound, this hybrid approach created erratic control behavior and actually worsened energy consumption compared to simpler strategies.

The parameter analysis showed that different cost functions respond differently to weight adjustments. Traditional tracking approaches showed little sensitivity to parameter changes, while economic formulations especially J_5. This suggests there's more optimization potential within economic frameworks.

While this study focused on the Fiat 500e model, the approach should work well for other electric vehicles. The basic tradeoffs between close following behavior and energy optimization remain relevant across different vehicle types, though specific parameter values would need adjustment.

Future work could explore more complex traffic scenarios involving multiple preceding vehicles or lane changes. Additionally, incorporating road grade information and more detailed battery models might further enhance energy optimization. Interesting extensions might include analyzing vehicles with dual energy sources, such as hybrid electric vehicles or fuel cell hybrid electric vehicles, where power management becomes even more nuanced due to multiple energy conversion pathways. Another promising direction would be studying platoon scenarios with multiple vehicles in series, where the energy dynamics of the entire system could be collectively optimized, potentially unlocking greater efficiency through coordinated control strategies.

Overall, my research shows that economic MPC offers a promising approach to extend the range of EVs during daily driving while maintaining appropriate driving behavior. The energy savings, although modest in percentage terms, could have a significant impact on the vehicle's range in normal use, and not on a single 30-minute simulation. The work done in this thesis shows how CTG principles can be successfully integrated into economic objectives for EV control, potentially finding applications in other domains such as hybrid powertrain configurations, smart grids, and HVAC systems.

Bibliography

[1] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[2] David Angeli, Rishi Amrit, and James B. Rawlings. On average performance and stability of economic model predictive control. *IEEE Transactions on Automatic Control*, 57(7):1615–1626, 2012.

[3] Hoseinali Borhan, Ardalan Vahidi, Anthony M Phillips, Ming L Kuang, Ilya V Kolmanovsky, and Stefano Di Cairano. MPC-Based Energy Management of a Power-Split Hybrid Electric Vehicle. *IEEE Trans. Control Syst. Technol.*, 20(3):593–603, 2011.

[4] 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. *IEEE Trans. Autom. Sci. Eng.*, 2025.

[5] European Parliament and Council. Regulation (EU) 2023/851 of the European Parliament and of the Council of 19 April 2023 amending Regulation (EU) 2019/631 as regards strengthening the CO2 emission performance standards for new passenger cars and new light commercial vehicles in line with the Union's increased climate ambition. *Official Journal of the European Union*, L 110:5–20, 2023.

[6] Antonio Ferramosca. Fundamentals of model predictive control: Economic

mpc. Lecture slides, Bergamo, 24/06/2022, June 2022.

[7] Giancarlo. Genta, Lorenzo. Morello, Francesco. Cavallino, and Luigi. Filtri. *The motor car : past, present and future*. Mechanical Engineering Series. Springer, Dordrecht [Netherlands, 1st ed. 2014. edition, 2014.

[8] G.C. Goodwin, M.M. Seron, and J.A. de Doná. *Constrained Control and Estimation: An Optimisation Approach*. Communications and Control Engineering. Springer, 2005.

[9] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Communications and Control Engineering. Springer, Cham, Switzerland, 2 edition, 2017.

[10] Defeng He, Jing Sun, and Wei Chen. Multiobjective economic mpc of constrained non-linear systems. *IET Control Theory & Applications*, 10(13):1487–1495, 2016.

[11] Hongwen He, Rui Xiong, Kai Zhao, and Zhentong Liu. Energy Management Strategy Research on a Hybrid Power System by Hardware-In-Loop Experiments. *Appl. Energy*, 112:1311–1317, Dec. 2013.

[12] The MathWorks Inc. Matlab version: 24.2.0.2806996 (r2024b), 2024.

[13] C. James and M. Zeilinger. Theory in model predictive control: Constraint satisfaction and stability, 2011. [Online]. Available: uiam.sk/pc11/data/workshops/mpc/MPC_PC11_Lecture1.pdf.

[14] Derick Furquim Pereira, Francisco da Costa Lopes, and Edson H Watanabe. Nonlinear Model Predictive Control for the Energy Management of Fuel Cell Hybrid Electric Vehicles in Real Time. *IEEE Trans. Ind. Electron.*, 68(4):3213–3223, 2020.

[15] Yuqi Qiu, Tao Zeng, Caizhi Zhang, Gucheng Wang, Yaxiong Wang, Zhiguang Hu, Meng Yan, and Zhongbao Wei. Progress and Challenges in Multi-Stack Fuel Cell System for High Power Applications: Architecture and Energy Management. *Green Energy Intell. Transp.*, 2(2), 2023.

[16] FCA Italy S.p.A. Nuova fiat 500 owner's manual. https://aftersales.fiat.com/eLumData/IT/00/332_NUOVA500/00_332_NUOVA500_603.85.158_IT_03_08.20_L_LG/00_332_NUOVA500_603.85.158_IT_03_08.20_L_LG.pdf, 11 2020. Document Number: 603.85.158, Accessed: 2025-02-15.

[17] Monica Tutuianu, Pierre Bonnel, Biagio Ciuffo, Takahiro Haniu, Noriyuki Ichikawa, Alessandro Marotta, Jelica Pavlovic, and Heinz Steven. Development of the world-wide harmonized light duty test cycle (wltc) and a possible pathway for its introduction in the european legislation. *Transportation research. Part D, Transport and environment*, 40:61–75, 2015.

[18] UNEP. Distribution of greenhouse gas emissions worldwide in 2023, by sector. https://www.statista.com/statistics/241756/proportion-of-energy-in-global-greenhouse-gas-emissions/, 10 2024. Accessed: 2025-02-10.

[19] UNRAE. Number of passenger cars sold in Italy from 2014 to 2021, by fuel type. https://www.statista.com/statistics/417567/passenger-car-sales-in-italy-by-fuel-type/, 1 2022. Accessed: 2025-02-12.

[20] Junmin Wang and R. Rajamani. Should adaptive cruise-control systems be designed to maintain a constant time gap between vehicles? *IEEE Transactions on Vehicular Technology*, 53(5):1480–1490, 2004.

[21] Weiwei Xin, Enyong Xu, Weiguang Zheng, Haibo Feng, and Jirong Qin. Optimal Energy Management of Fuel Cell Hybrid Electric Vehicle Based on Model Predictive Control and On-Line Mass Estimation. *Energy Rep.*, 8:4964–4974, Nov. 2022.

[22] Wenbin Zhang, Jianqiu Li, Liangfei Xu, and Minggao Ouyang. Optimization for a Fuel Cell/Battery/Capacity Tram with Equivalent Consumption Minimization Strategy. *Energy Convers. Manag.*, 134:59–69, Feb. 2017.