CHALMERS

Development of energy-optimal control strategies for a fully electric vehicle.

Master of Science Thesis in the Master's Programme Automotive Engineering

ALEJANDRO FERREIRA PARRILLA

Department of Automatic Control
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2014
Master's Thesis 2014:EX050

MASTER'S THESIS 2014:EX050

Development of energy-optimal control strategies for a fully electric vehicle.

Master of Science Thesis in the Master's Programme

ALEJANDRO FERREIRA PARRILLA

Department of Automatic Control
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2014

Development of energy-optimal control strategies for a fully electric vehicle

Master of Science Thesis in the Master's Programme Engineering
ALEJANDRO FERREIRA PARRILLA

Master thesis / Department of Signals and Systems, Automatic Control group,
Chalmers University of Technology 2014:EX050

Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: + 46 (0)31-772 1000

Department of Signals and Systems, Göteborg, Sweden 2014

Development of energy-optimal control strategies for a fully electric vehicle.

Master of Science Thesis in the Master's Programme
ALEJANDRO FERREIRA PARRILLA
Department of Automatic Control
Chalmers University of Technology

ABSTRACT

The development of new strategies in order to increase the range of fully electric vehicles by minimizing the energy consumption using on-board and off-board sources has been the subject of this work. To do so, an innovative energy management strategy has been developed for intersections regulated by Traffic Lights.

EU founded research projects like OpEneR unify efforts of several partners to develop such new technologies. This work contributes to OpEneR project and deals with Traffic Lights. The objective is to develop energy optimal control strategies for the fully electric OpEneR prototypes. It is assumed that Vehicle-to-Infrastructure (V2I) communication provides required data about Traffic Lights along considered road segment. Model Predictive Control is used as the framework methodology, which allows for specific powertrain representation and managing of the Traffic Light problem through constraints. In order to test the efficiency of developed control method, the controller is subjected to several representative simulation use-cases, including a real world road in Graz, Austria. The results have shown energy consumption reductions compared to a standard driver varying between 17 and 37 % depending on selected use-case. It is therefore concluded that the presented methodology constitutes an effective approach to the Traffic Light control problem management.

Additionally, a Speed Advisory optimization-free algorithm dealing with Traffic Lights was also introduced. It is designed as a real-time implementable alternative to Model Predictive Control formulation. Furthermore, work performed with Human Machine Interface as well as a vehicle mock-up for demonstration purposes are also described. Finally, investigations regarding Explicit and Nonlinear Model Predictive Control are also presented and their strengths and drawbacks are highlighted.

The Master Thesis has been written in close collaboration with AVL List GmbH and contributes to the European research project OpEneR.

Key words: Traffic Lights, Model Predictive Control (MPC), Electric Vehicle, Energy consumption, Advanced Driver Assistance Systems (ADAS), Vehicle-to-Infrastructure (V2I) Communication, Speed Advisory, Vehicle Simulator.

Development of energy-optimal control strategies for a fully electric vehicle.

Master of Science Thesis in the Master's Programme

ALEJANDRO FERREIRA PARRILLA
Department of Automatic Control
Chalmers University of Technology

CONTRIBUTIONS

The work described in this thesis has also contributed with content in the following publications:

- E.Kural, A.Ferreira Parrilla, S.Jones, A.Grauers, "Traffic Light Assistant System for Optimized Energy Consumption in an Electric Vehicle". IEEE ICCVE Connected vehicles, Vienna, 2014.

- Patent application on Traffic Light Assistant "System and a method for operating an ego vehicle with at least one prime mover on a route during approaching and passing one or several traffic lights". Patent nr. 14167936.5-1803.

Contents

1 INTRODUCTION 1

1.1 OpEneR project 1

1.2 Complex Simulation Tool-Chain 2

1.3 Model Predictive Control. An Introduction. 3

1.4 State of the Art. 5

2 PROBLEM FORMULATION UNDER MPC FRAMEWORK 7

2.1 Plant modelling 7

2.2 Prediction model 9

2.3 Cost Function 10

2.4 System Constraints 12

2.5 MPC structure for Traffic Light problem 15

2.6 Traffic ahead consideration 17

2.7 Complete Mathematical Problem Definition 19

3 IMPLEMENTATION CHALLENGES 21

3.1 Controller development in Matlab/ Simulink 21

3.2 Pedal inputs corresponding to acceleration control 23

3.3 Cost function relative weighting factors tuning 24

3.4 Relative tuning of Control and Prediction horizons 24

3.5 Tuning of revaluation frequency and vehicle ahead prediction horizon 25

4 RESULTS AND USE-CASES 28

4.1 Use-case 1: Single Traffic Light 29

4.2 Use-case 2: Two Traffic Lights 30

4.3 Use-case 3: Three Traffic Lights and Vehicle ahead. 31

4.4 Use-case 4: Graz route segment with multiple Traffic Lights 33

5 SPEED ADVISORY 37

5.1 Integration to Human Machine Interface software 41

5.2 Integration to Mock-up. 43

6 PARALLEL INVESTIGATIONS 46

6.1 Explicit MPC 46

6.2 NonLinear MPC 49

7 CONCLUSION AND FUTURE WORK 52

8 REFERENCES 54

Preface

In this study, optimal control strategies have been developed for fully electric vehicle. The project has been conducted at the AVL facilities in Graz, Austria. The work is a part of a European research project, OpEneR, aiming at developing driving strategies and driver assistance systems for increased driving range and safety. The developed technologies are implemented on a fully electrical passenger vehicle .The project is founded under the Seventh Framework Program, and the support is gratefully acknowledged.

This part of the project has been carried out by student Alejandro Ferreira Parrilla with Emre Kural (AVL Graz) and Associate Professor Anders Grauers (Chalmers University of Technology) as supervisors.

Graz, September 2014.

Alejandro Ferreira Parrilla

Notations

$4WD$	Four Wheel Drive
$ADAS$	Advanced Driver Assistance Systems
ACC	Adaptive Cruise Control
CAN	Control Area NetworkDP Dynamic Programming
EM	Electric Machine
$EMPC$	Explicit Model Predictive Control
GPS	Global Positioning System
HiL	Hardware in the Loop
HMI	Human Machine Interface
ITS	Intelligent Transportation System
MiL	Model in the Loop
ML	Matlab
MPC	Model Predictive Control
$MPT3$	Multi Parametric Toolbox
$OpEneR$	Optimal Energy consumption and Recovery –7th framework EU project
PWA	Piece-Wise Affine
QP	Quadratic Programming
SA	Speed Advisory
SiL	Software in the Loop
SL	Simulink
SoC	State of Charge
TL	Traffic Light
TLA	Traffic Light Assistant
VCU	Vehicle Control Unit
$V2I$	Vehicle-to-Infrastructure

List of Figures

Figure 1: OpEneR 4WD Twin Axle Fully Electrical Vehicles....2

Figure 2: Simulation tool-chain composed of AVL CRUISE, ML/SL, IPG CARMAKER....3

Figure 3: Schematics of MPC controller structure....4

Figure 4: 4WD Fully Electrical OpEneR vehicle model in AVL CRUISE....7

Figure 5: Schematic of Traffic Light Assistant (TLA). 13

Figure 6: Generic diagram representing the feasible position regions in future time instants. 14

Figure 7: Vehicle's acceleration-speed capability based upon EM(s) characteristics..15

Figure 8: Gap selection process in high-level Fast MPC....17

Figure 9: Fast MPC integration to ensure most optimal TL gap selection for all time instants 17

Figure 10: Feasible region modification due to preceding vehicle imposing new set of constraints. 19

Figure 11: Simulink implementation of MPC controller....23

Figure 12: Detailed view of PI for Gas/Brake pedals from desired acceleration 24

Figure 13: Problem turned infeasible due to too large vehicle ahead prediction horizons.....26

Figure 14: Problem turned infeasible due to not frequently enough gap revaluation..26

Figure 15: Trajectories for MPC and Normal Driver vehicles in use-case 1....29

Figure 16: Battery SoC figure for MPC and Normal Driver vehicles in use-case 1 ...30

Figure 17: Trajectory plots for MPC and Normal Driver vehicles in use-case 2 31

Figure 18: SoC evolution for MPC and Normal Driver vehicles in use-case 2....31

Figure 19: Trajectory plots for MPC, Normal Driver and preceding vehicle in use-case 3....32

Figure 20: SoC evolution for MPC and Normal Driver in use-case 3....33

Figure 21: Google Maps representation of selected Graz road....34

Figure 22: Trajectories for MPC and Normal Driver vehicles in Graz route 35

Figure 23: SoC evolution for MPC and Normal Driver vehicles in Graz route 35

Figure 24: Schematic representation of deviation from initial trajectory criteria. Green line represents unaltered vehicle trajectory.....38

Figure 25: Delta time between two selected gaps.....38

Figure 26: Quick crossing trajectories against minimal deviation from initial kinematics trajectory. 39

Figure 27: Exemplification of gaps available for a scenario with 3 TLs with 3 gaps each. 40

Figure 28: Cost to go matrix representation. Rows represent starting gap, while columns represent end gap. 40
Figure 29: Trajectory plots for MPC, Speed Advisory and Normal driver vehicles ... 41
Figure 30: SoC plots for MPC, Speed Advisory and Normal Driver vehicles 41
Figure 31: Concept HMI for OpEneR prototypes developed by CTAG. Custom icons for TL scenario management. 42
Figure 32: Layout for HMI testing using single simulation PC 43
Figure 33: Structure for RT HiL Mock-up setup. 44
Figure 34: Detail of Mock-up HMI and HuD 44
Figure 35: Mock-up setup at CTAG 45
Figure 36: TLA interfaces to Driver 45
Figure 37: Polyhedral State Space partition in Critical Regions. 48
Figure 38: Optimal control inputs for each different Critical Region. 48
Figure 39: Schematic for possibilities to deal with nonlinear MPC 50

List of Tables

Table 1: Values of cost function integration over time for different values of Control Horizons for a specific manoeuvre. (N_p=30 for all cases) 25
Table 2: Traffic Light data for use-cases 1 to 3 28
Table 3: Initial kinematic quantities for use-case 1 29
Table 4: Initial kinematic quantities for use-case 2 30
Table 5: Initial kinematic quantities for use-case 3 32
Table 6: Traffic Light data for selected route segment in Graz 34
Table 7: Energy and time savings for selected use-cases 36
Table 8: Traffic Light data for selected manoeuvre 47

1 Introduction

Intelligent vehicle driving systems are nowadays becoming a necessity for sustainable and safe mobility in the context of modern society. As the world population is moving towards city living, sustainable and green mobility plays an essential role for such areas. To a great extent individual mobility relies on traditional combustion passenger cars, but the imminent shortage of fossil fuels is pushing academia and industry towards other alternatives such as hybridization or electrification. In that regard, electric mobility is especially suited for city driving, where the travelled distances are relatively short and the constant stop and go manoeuvers allow for regenerative braking [1].

Traffic lights (TL) coordinate the traffic for most of the city intersections. A busy intersection in a typical urban area might coordinate the movement of thousands of vehicles a day [2]. In this context, the cumulated energy waste due to stopping at red light and the corresponding generated emissions become crucial. Therefore, being able to fluently cross the Traffic Lights is of great value. A Traffic Light Assistant (TLA) would therefore contribute to improve the overall transportation system energy efficiency and, for the case of a fully electric vehicle, to the all-electric range extension. It is well known that electric vehicles suffer problems of limited range.

In this Thesis, it is assumed that the electric vehicle -the OpEneR prototype- , has advanced information access thanks to on-board systems as well as Vehicle-to-Infrastructure (V2I) communication. I.e. information about Traffic Light shifting times, distances to TL, ego-vehicle position, or other vehicle's kinematics can be obtained in real time and are accessible to the Vehicle Control Unit (VCU).

This thesis is written in close collaboration with AVL List in Graz, Austria, as part of the OpEneR project and its purpose is to develop optimal control methods to deal with Traffic Light scenarios. The goal is to develop control algorithms that can reduce energy consumption required to cross upcoming Traffic Lights.

1.1 OpEneR project

OpEneR (Optimal Energy consumption and Recovery), is a European research project launched in May 2011 under the Seventh Framework Program (grant agreement n. 285526). The aim of the project is to develop advanced driving strategies and assistance systems which increase vehicle efficiency and therefore range. The project involves the following institutions:

- Robert Bosch GmbH
- Robert Bosch Car Multimedia GmbH
- Peugeot Citroën Automobiles S.A.

- AVL List GmbH
- Centro Tecnológico de Automoción de Galicia
- FZI Forschungszentrum für Informatik an der Universität Karlsruhe

During the development of the project, several prototype vehicles were built to ultimately test developed technologies (Figure 1). The OpEneR prototypes are 4WD full electric vehicles with two permanent magnet 50 kW electric machines, connected through a dog clutch to front and rear axles respectively. The base vehicle is the Peugeot 3008 Hybrid4. The electric energy is supplied by a 40 kWh Li-ion battery package and the estimated range for the prototypes is approximately 200 km. [3]

Figure 1: OpEneR 4WD Twin Axle Fully Electrical Vehicles

1.2 Complex Simulation Tool-Chain

In order to test the control strategies developed within this master project, an accurate and reliable simulation platform that faithfully represents all relevant aspects of vehicle, its environment and driver is a need. A powerful simulation tool-chain composed of IPG CarMaker, AVL CRUISE and Matlab/Simulink is used for this work. The controllers are developed in Matlab/Simulink environment, while as vehicle and realistic traffic environment are provided by IPG CarMaker. Specific powertrain configurations are managed by the in-house software AVL CRUISE.

The Co-simulation toolchain combining the above mentioned enhanced simulation tools is shown in Figure 2, and it supports complex vehicle and powertrain development. It uses highly developed simulation tools for each specialized application. Integration of diverse tools is set up through standard interfaces and yields an enhanced model-in-the-loop (MiL) and Software-In-The-loop (SiL) platform. This platform is also extendable to Real-time on a Hardware-in-the-Loop (HiL) or Powertrain Testbed.

Besides an accurate simulation model, the ease of generating complex traffic scenarios and 3D modelling of the test roads is a major feature of the developed simulation platform. OpEneR technologies can be tested on a dedicated public road test corridor in Vigo, Spain, which features enhanced digital maps, and a real car-to-car and car-to-infrastructure communication network operated by CTAG. Detailed road information for the Vigo test corridor (e.g. curvatures, inclination, speed limits) is included in the simulation toolchain for offline development and pre-calibration of the control strategies. The benefit of this approach is that the developed control algorithms are tested on realistic real world driving routes and real world benefits in terms of energy efficiency are determined early in the development process.

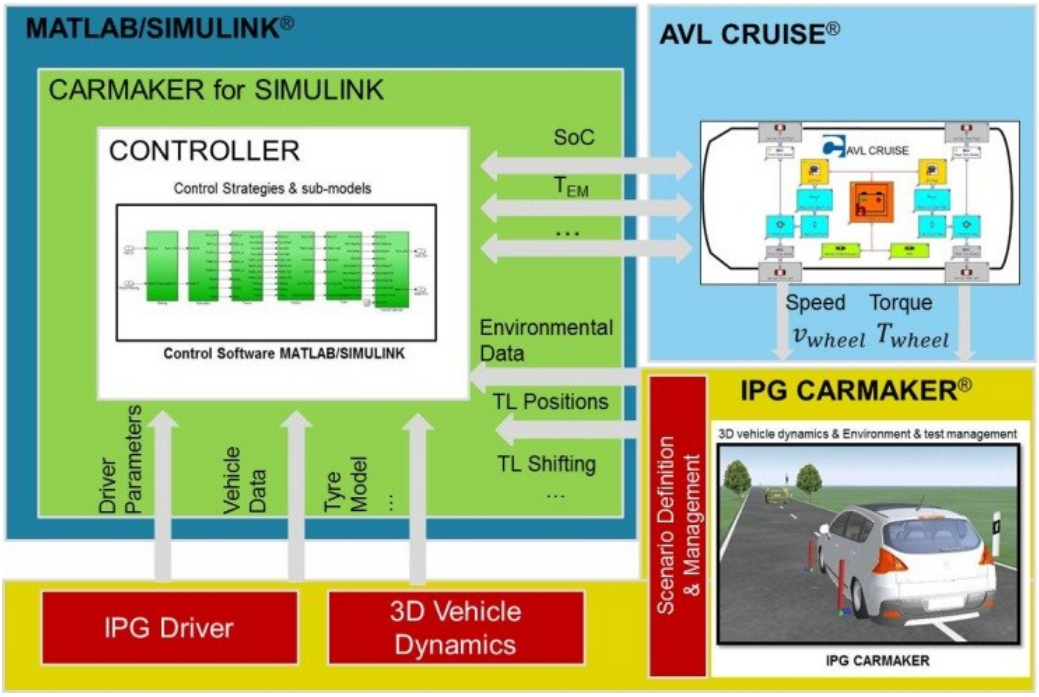

Figure 2: Co-simulation toolchain consisting of AVL CRUISE, IPG CarMaker®, and MATLAB/Simulink®

1.3 Model Predictive Control. An Introduction.

A vehicle integrated in an Intelligent Transportation System (ITS) is to receive information about upcoming road segments. This vast amount of data can range from upcoming speed limits, traffic conditions ahead, to gradients or Traffic Light status. This enables the vehicle to look ahead several kilometres along the road and opens multiple possibilities to optimize energy consumption. Model Predictive Control (MPC) is often cited as one of the most popular advanced control techniques. MPC was firstly used in Chemical industry, but in recent time the Automotive industry has shown growing interest for MPC applications including engine, transmissions, emissions, energy management, etc. [4]. The main reasons for its success are that MPC:

- Handles multivariate problems with ease.
- Can take into account actuator limitations.
- Plant physical constraints are intuitively implemented to the formulation. [5]

On the other hand, main drawback for MPC is the required online optimization. In case of large plant models, the required calculations can quickly increase. This needs special care in case of systems with fast dynamics. It must also be noted that in the case of non-Linear systems, the optimization problem to solve each time-step becomes non-linear and difficulties in solving the constrained optimization problem arise.

In essence, MPC is based on optimizing the future plant control trajectory subject to constraints. Those constraints can be classified under plant states constraints, plant control constraints, or plant output constraints. (Note that the constraints can be functions of time). In traditional MPC formulation, the objective consists of minimizing a given cost function *J*. This objective or cost function penalizes deviations of states from reference set-points while limiting control action to do so. The optimization problem is often a constrained optimization problem, and ease to include a large variety of constraints to the formulation is one of the main features of MPC. The result of the optimization process is the optimal control sequence U_{opt} for the control horizon $N_{c,}$. Only the first element of the optimal control sequence is applied to the plant. The rest of the control sequence is employed for plant behaviour estimation in the prediction module. This optimization is repeated each time-step in a receding horizon fashion. [5-7]. A schematic representation of MPC structure can be found in Figure 3. Interested readers are referred to [5-7] for more detailed descriptions.

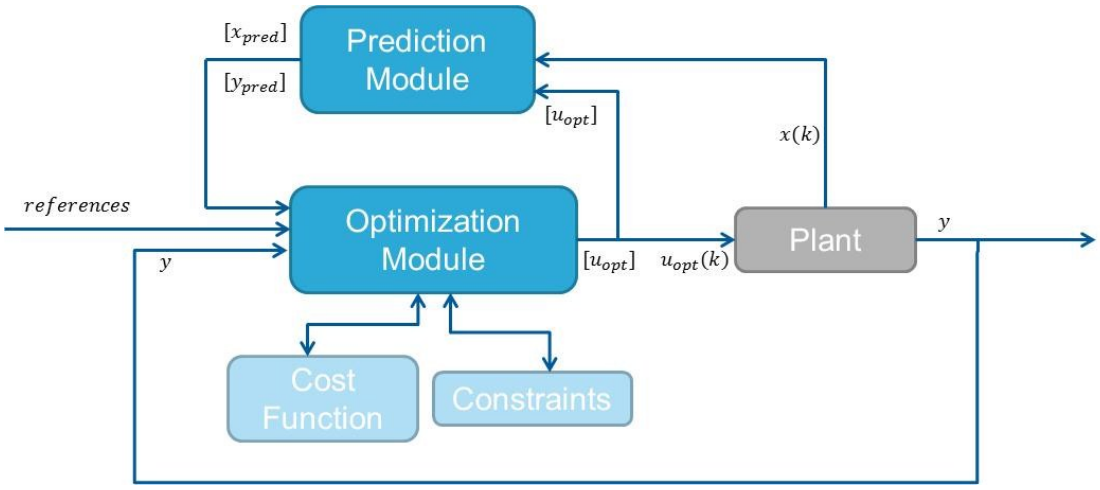

Figure 3: Schematics of MPC controller structure

As it will be shown in Chapter 2, the proposed MPC formulation includes the following factors for handling the Traffic Light problem:

- Repeated calculation of the upper-level "Green-Wave" optimal trajectory, which accounts for changes in traffic conditions.
- A lower-level, high-precision MPC for more detailed trajectory calculations.
- Ego-vehicle dynamic behaviour as well as acceleration and deceleration limits.
- Prediction of the trajectory of the vehicle ahead.

1.4 State of the Art.

This section presents state of the art techniques extracted from literature to deal with Traffic Lights. In addition, contributions of this thesis will be differentiated from already existing publications.

Institutional pressure on reduced emissions and fuel consumption has encouraged industry and research institutes to produce innovative advanced vehicle driving strategies. Numerous publications around this topic have seen the light of day. Publications [8-11] provide good examples of nowadays trends for electric vehicles and connected powertrain.

Some studies regarding optimized efficiency for vehicles in the context of Traffic Lights can be found in the literature. They can mainly be divided into two categories: macroscopic traffic signal control system and microscopic speed modulation techniques.

Traffic lights are centrally operated, and their shifting schedule is constantly updated according to time of the day, predicted traffic, weather forecasts, etc. Such macroscopic systems are generally statistically optimized and centrally controlled, and allow for a "Green Wave" like behaviour: traffic flow is maximized and the Traffic Lights are coordinated such that vehicles can avoid stopping as much as possible by catching the so called "Green Wave". [2]

On the other hand, microscopic -from an ego-vehicle stand point- speed modulation techniques have also been proposed in the literature. State of the Art speed modulations techniques such as the GLOSA (Green Light Optimal Speed Advisory) initiative [13] provide the driver with a global optimal speed recommendation. Other papers derive constant speed trajectories in order to cross TL during green phases [13]. Nevertheless, such speed advisory systems do not take into account specific powertrain configuration and therefore are not optimal for each individual vehicle. In general, the generated speed recommendations are calculated off-board and broadcasted to the whole traffic. The proposed Traffic Light Assistant developed under MPC framework is an on-board feature that has the potential to be adapted to any specific powertrain.

The Thesis is structured as follows:

The Traffic Light problem handling under Model Predictive Control framework is detailed in Chapter 2. This section represents the main contribution and core of this Thesis. To do so, firstly, a model of the vehicle for control purposes is derived. Subsequent sections present prediction model and the selected cost function. Sets of constraints are then added to the formulation in order to deal with Traffic Lights, powertrain limitations but also other vehicles. Finally, the chosen parallel MPC structure to deal with the whole problem is described.

Some details about the controller implementation in the simulation tool-chain as well as some practical issues when it comes to the implementation of the control strategies are presented in Chapter 3. Details on how controllers are migrated from Matlab to Simulink environment are given and implementation challenges are highlighted. Tuning of controller parameters are of special importance. The effects of different values for controller parameters are explained in a comprehensive manner.

Chapter 4 thereafter presents the results for some relevant use-cases. Selected cases aim at presenting the results in a progressive manner with increasing control complexity: from a route with a single Traffic Light to a Multiple Traffic Light road segment with traffic ahead. At the end of this Chapter, realistic road data for a route section in Graz is implemented in the simulation environment for testing of the control strategies.

Chapter 5 presents an alternative formulation to Model Predictive Control for Traffic Light problem handling. The generated control strategy, named Speed Advisory, offers a less computationally demanding alternative, while providing more than satisfying results. The developed Speed Advisory is then integrated with Human Machine Interface software which is part of OpEneR project. As a final step towards real in-vehicle implementation, the developed software is taken to the Real-Time environment and tested in OpEneR mockup at CTAG in Vigo, Spain.

Chapter 6 offers a quick overview of parallel investigations carried out under the scope of this Thesis. The idea behind it is to investigate alternatives to standard MPC formulation that could be applicable to Traffic Light and other problems. More precisely, some considerations about Explicit and Nonlinear MPC and their applicability to the Traffic Light problem are given.

Finally, main conclusions are drawn in Chapter 7 and possible future investigation directions are also outlined.

2 Problem formulation under MPC framework

The typical scenario considered in this thesis consists of a vehicle traveling along the road and approaching a group of Traffic Lights. Such scenario can be easily simulated with the aforementioned tool-chain. Nevertheless, for control strategies to be effective there must be suitable good mathematical formulation of the use-case in the controller. Therefore, one of the main challenges arising is to couple formulation imposed by MPC methodology with the particular type of Traffic Light scenarios.

This section will present the chosen approach to deal with the Traffic Light problem in Model Predictive Control framework.

2.1 Plant modelling

An adequate plant model is the foundation for any control strategy aiming at controlling a dynamic system. The vehicle dynamics are inherently non-linear, and the plant modelling becomes complex. Since MPC solves an optimization problem in real-time, the plant model is limited in complexity. Real vehicle dynamics would imply solving non-linear optimization problem. Therefore, a simplistic model of the vehicle is chosen. This represents a typical trade-off between accuracy and real-time implementation. The underlying complex vehicle powertrain model is briefly described in Figure 4 and is providing realistic representation of the physical prototypes. The accuracy and validity of this model has been proven, but its inherent complexity is further simplified for MPC purposes.

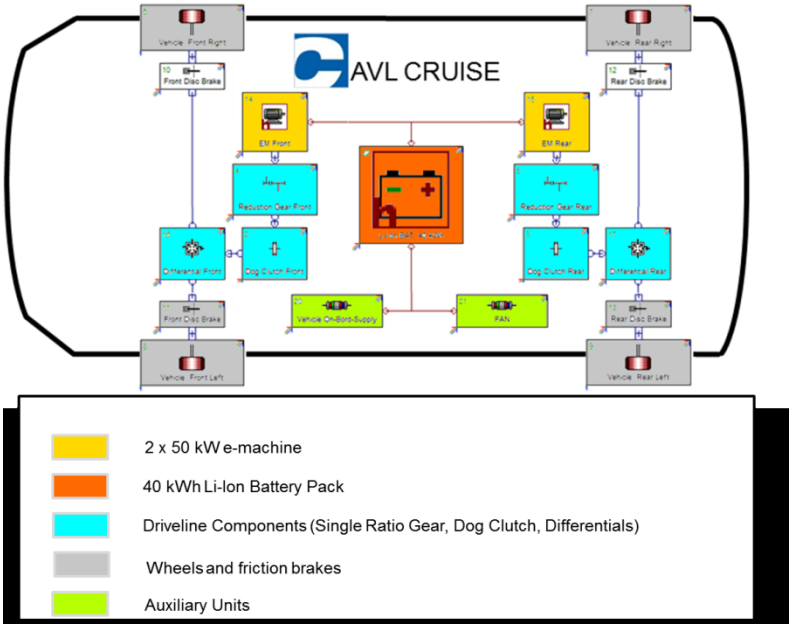

Figure 4: Powertrain layout of 4WD Twin Axle Fully Electrical OpEneR Vehicle.

To simplify matters it is reasonable to neglect lateral as well as pitch dynamics since neither of them has a significant impact on energy consumption. The longitudinal dynamics of the vehicle are expressed as basic kinematic relationships together with some dynamic acceleration constraints. Dynamic acceleration constraints are defined as limits in vehicle longitudinal acceleration as function of vehicle speed (See Section 2.4.2).

The linear model representing the vehicle is derived from the equations describing how position, *p*, speed, *v*, and acceleration, *a*, are related:

$$\mathrm{v} = \frac{\mathrm{dp}}{\mathrm{dt}} \tag{1}$$

$$a = \frac{dv}{dt} \tag{2}$$

Equation (3) approaches real vehicle inertial longitudinal dynamics by a first order system. A transfer function between desired acceleration, a_{des} and real acceleration of the vehicle is implemented.

$$\dot{a} * T_f + a = K_f * a_{des} \quad \rightarrow \quad a = \frac{K_f}{T_f * s + 1} * a_{des} \tag{3}$$

The system gain, K_f and the time constant T_f are chosen appropriately to represent vehicle´s dynamic behaviour and are estimated via a kick-down test. The reason behind this first order dynamics is that acceleration changes do not occur instantly in the real vehicle due to inertial effects.

The equations above are formulated under compact state-space representation. System states are expressed as the state vector $x = [p, v, a]$. The system control variable $u = a_{des}$ represents the vehicle desired longitudinal acceleration. The system output, *y*, is chosen to be position since it eases reference setting (see later sections). The continuous time state space representation of the vehicle movement then can be expressed as

$$\dot{x} = A * x + B * u \tag{4}$$

$$y = C * x \tag{5}$$

With

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \frac{-1}{T_f} \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ \frac{K_f}{T_f} \end{bmatrix} \quad C = [1 \quad 0 \quad 0] \tag{6}$$

The continuous time representation described in equations (1-3), needs to be adapted to discrete time for simulation and since MPC is using a time discrete formulation. This is done via a zero order hold discretization with sample time T_s. For most of the simulations, a value of $T_s = 1\,sec$ was selected as a reasonable trade-off between accuracy and calculation time.

2.2 Prediction model

The above described vehicle model constitutes the core of MPC algorithm. MPC solves an optimization problem every time step. The result is an optimal control sequence. This control sequence is employed together with the vehicle model to predict system behaviour along the selected prediction horizon N_p. Predicted future system states are calculated with previous time-step optimal control sequence and actual states from plant measurements, $x(k_i)$ (Figure 3). The general derivation for predicted states reads:

$$x(k_i + 1) = A * x(k_i) + B * U(k_i) \tag{7}$$

$$x(k_i + 2) = A * x(k_i + 1) + B * U(k_i + 1) \tag{8}$$

If equation (7) is substituted into equation (8),

$$x(k_i + 2) = A * \big(A * x(k_i) + B * U(k_i)\big) + B * U(k_i + 1) = A^2 * x(k_i) + A * B * U(k_i) + B * U(k_i + 1) \tag{9}$$

The derivation can be easily extended for outputs considering eq. (5). Extending the analysis to the whole prediction horizon and presenting it in a more compact way (note that the notation will now assume that the control sequence is the optimal control sequence obtained in the previous time-step, $u_{opt,prev}$):

$$x_{pred} = M * x(k_i) + N * u_{opt,prev} \tag{10}$$

$$y_{pred} = C * x_{pred} = E * x(k_i) + F * u_{opt,prev} \tag{11}$$

Where,

$$x_{pred} = \begin{bmatrix} x(k_i+1) \\ x(k_i+2) \\ \dots \\ x(k_i+N_p) \end{bmatrix} \quad u_{opt,prev} = \begin{bmatrix} u(k_i+1) \\ u(k_i+2) \\ \dots \\ u(k_i+N_c) \end{bmatrix} \tag{12}$$

$$M = \begin{bmatrix} A \\ A^2 \\ \dots \\ A^{N_p} \end{bmatrix} \quad N = \begin{bmatrix} B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & \dots & A^{N_p-N_c}B \end{bmatrix} \tag{13}$$

$$E = \begin{bmatrix} CA \\ CA^2 \\ \dots \\ CA^{N_p} \end{bmatrix} \quad F = \begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & \dots & CA^{N_p-N_c}B \end{bmatrix} \tag{14}$$

2.3 Cost Function

As mentioned previously, MPC formulation includes the minimization of a certain functional in every time-step. Standard cost functions are usually composed of two terms: the first term penalizes deviations of system outputs from a reference set-point chosen by the engineer. The second term penalizes amplitude of control action to bring the plant to the states reference (eq. 15). Therefore, when applying this standard formulation to the considered vehicle in Traffic Light scenario, it is clear that cost function needs to aim at minimizing energy consumption of the vehicle. The approach taken in this Thesis is to try to minimize vehicle speed deviations from its initial value. In other words, constant speed trajectories are preferred whenever possible. This naturally translates into a minimization of required vehicle acceleration (first term of eq.15). Additionally, by selecting an appropriate position reference set-point, generated trajectories can be tuned to adjust total vehicle travel time as explained later.

Typically the cost function, J_i, is quadratic in the set-point error and system output and can then be expressed as

$$J_i(k_i) = \sum_{t=k_i}^{N_p-1} U^T R U + \left(y_{pred} - y_{ref}\right)^T Q \left(y_{pred} - y_{ref}\right) \tag{15}$$

where Q and R are tuning matrices which weight the different terms in the cost function. Tuning of such parameters is of vital importance to obtain the correct controller behaviour. (See Chapter 3, implementation challenges).

The following section will prove that the chosen cost function can be represented as function of only the control input *u* and that the final form is a quadratic cost function.

Firstly, let´s recall the fundamental equation of the prediction module:

$$y_{pred} = E * x(k_i) + F * U_{opt,prev} \tag{16}$$

By substitution of this equation into (15),

$$J_i(k_i) = \sum_{t=k_i}^{N_p-1} U^T R U + \left(E * x(k_i) + F * u_{opt,prev} - y_{ref}\right)^T Q (E * x(k_i) + F * \\ * u_{opt,prev} - y_{ref}) \tag{17}$$

If eq. (17) is further worked,

$$J_i(k_i) = \sum_{t=k_i}^{N_p-1} y_{ref} * Q * {y_{ref}}^T + \left(E * x(k_i) - 2 * y_{ref}\right) * Q * \left(E * x(k_i)\right) + 2 \\ * U * (k_i) * N^T * Q * \left(E * x(k_i)\right) + U * (F^T * Q * F + R) * U^T \\ = cte + G * U + U * H * U \tag{18}$$

It can be seen that the selected cost function is quadratic in *U* and therefore convex. This has numerous advantages. Especially important is the ease and speed in solving it. MATLAB solver QUADPROG can solve quadratic functions subject to constraints. [12]

As mentioned previously, the reference set-point in the cost function is used to adjust total vehicle travel time. The reference set-point, y_{ref} is changing depending on the scenario and Traffic Light configuration. For instance, let's consider a scenario with a single Traffic Light. In that case, the reference set-point would be the absolute Traffic Light position along the road. Minimization of differences of real vehicle position with respect to that reference trajectory will therefore result in control inputs aiming to decrease travel time. In other words, by including the position of the targeted Traffic Light in the cost function, solutions can be pushed towards reaching the Traffic Light quicker. This selected cost function together with the set of constraints defined in next section aim at minimizing vehicle energy consumption while respecting constraints introduced by the Traffic Lights shifting.

2.4 System Constraints

One of the main advantages of MPC is the ease to include constraints in the formulation. The main challenge of this Thesis was to find the correct formulation to the problem imposed by upcoming Traffic Lights. The selected approach is relying on constraints to restrict the feasible regions for the vehicle to be in, since Traffic Lights are objects that impose constraints on vehicle positions over time. This is done through constraints on system states. As detailed in section 2.4.2, feasibility of the generated control inputs for the real vehicle is also guaranteed thanks to additional set of constraints, so that the controller does not require accelerations that are not feasible for the real vehicle.

2.4.1 Constraints on system states

An upcoming series of Traffic Light poses constraints on vehicle position at given time instants: the position of the vehicle must not exceed TL position while its status is red. Similarly, the vehicle is free to cross the TL when its phase is green:

A vehicle with initial velocity v_0 is approaching a group of Traffic Lights at positions x_{TL1}, x_{TL2}…etc. It aims at crossing the aforementioned TL under green phase. These phases are characterized by $[t_{gi} < t < t_{ri}]$ where t_{ri} denotes green to red switching time and t_{gi} red to green switching time. The quantities are also defined in Figure 5. It is assumed that the Vehicle Control Unit (VCU) is aware of those quantities thanks to V2I communication.

Such time dependant constraints naturally transition to constraints on predicted states of the system. In other words, the vehicle position has to be in predefined and constrained regions between Traffic Lights as shown in Figure 6. In a more mathematical description these constrained regions are defined by:

$$\begin{aligned} p(k) \in R_I \quad if \quad 0 < t < t_{g1} \\ p(k) \in R_{II} \quad if \quad t_{g1} < t < t_{r1} \\ p(k) \in R_{III} \quad if \quad t_{r1} < t < t_{g2} \\ p(k) \in R_{IV} \quad if \quad t_{g2} < t < t_{r2} \end{aligned} \tag{19}$$

...

Speed constraints are also added to the formulation: for all time instants the speed cannot be negative:

$$v(k) > 0 \qquad if \qquad t > 0 \tag{20}$$

Previously mentioned constraints can be extended to the whole prediction horizon, N_p as upper and lower limits for the system states for future time instants:

$$[x_{min}] = \begin{bmatrix} p_{min}(k_i+1) \\ v_{min}(k_i+1) \\ a_{min}(k_i+1) \\ \vdots \\ p_{min}(N_p) \\ v_{min}(N_p) \\ a_{min}(N_p) \end{bmatrix} \leq \begin{bmatrix} p_{pred}(k_i+1) \\ v_{pred}(k_i+1) \\ a_{pred}(k_i+1) \\ \vdots \\ p_{pred}(N_p) \\ v_{pred}(N_p) \\ a_{pred}(N_p) \end{bmatrix} \leq \begin{bmatrix} p_{max}(k_i+1) \\ v_{max}(k_i+1) \\ a_{max}(k_i+1) \\ \vdots \\ p_{max}(N_p) \\ v_{max}(N_p) \\ a_{max}(N_p) \end{bmatrix} = [x_{max}] \tag{21}$$

Note that constraints on predicted system acceleration are similar to the control input constraints described in next section.

In a more compact form:

$$\begin{bmatrix} N \\ -N \end{bmatrix} * U \leq \begin{bmatrix} [x_{max}] - M * x(k_i) \\ -[x_{min}] + M * x(k_i) \end{bmatrix} \tag{22}$$

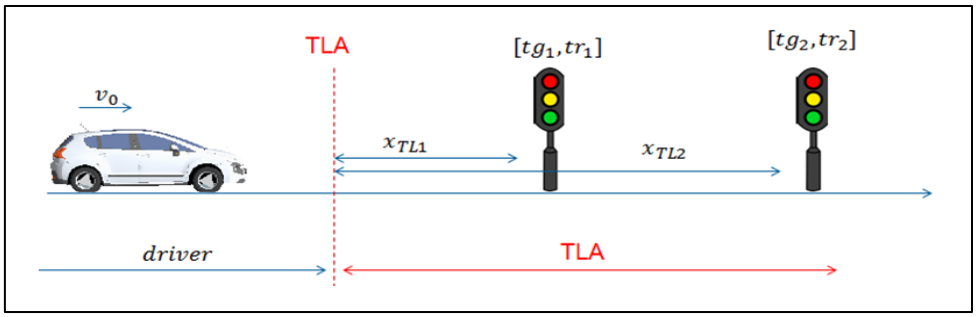

Figure 5: Schematic of Traffic Light Assistant (TLA).

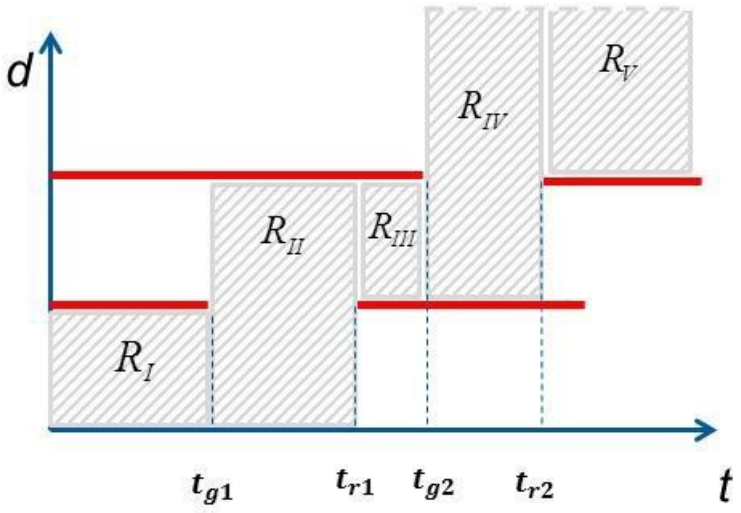

Figure 6: Generic diagram representing the feasible position regions in future time instants.

2.4.2 Constraints on control inputs

Besides system states constraints, it is necessary to ensure that the controller does not request unrealistic accelerations which would be infeasible by the real vehicle. To do so, the control inputs as well as system states are restricted. This means that, at each time-step, the optimal control sequence generated by the optimization solver (desired acceleration, a_{des}) is subjected to constraints.

As described previously, OpEneR prototypes have two electric machines. Since intervention of mechanical friction brakes would lead to kinetic energy being wasted as heat instead of regenerated and stored in the battery, the control inputs should remain within the recuperation potential if the manoeuvre allows so. Similarly, requested control inputs must not exceed the EM's capabilities in forward acceleration. Such limits are changing dynamically with vehicle longitudinal speed. The limits for the OpEneR vehicle are shown in Figure 7. In the figure one can easily recognize the shape of EM standard ω-T curves. Such curves were then transformed to vehicle speed- vehicle acceleration curves through specific testing with the vehicle model. Gas pedal was set to its maximum value and acceleration vs speed datapoints were collected. Brake pedal was similarly actuated from maximum speed with disabled mechanical braking to obtain the full recuperation curves. The resulting data, once conveniently filtered, can be seen below.

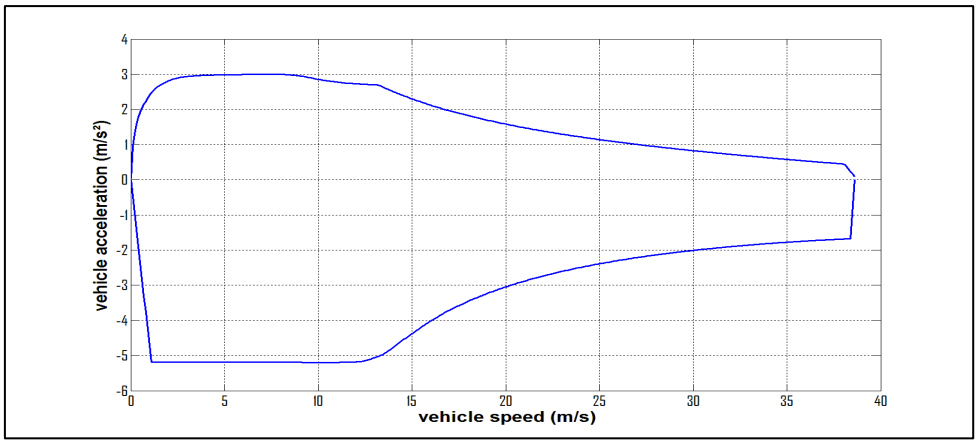

Figure 7: Vehicle's acceleration-speed capability based upon EM(s) characteristics

As described in Figure 7, control input constraints are a function of vehicle speed:

$$U_{min}(v) \leq U \leq U_{max}(v) \tag{23}$$

As a consequence, those constraints will presumably be different for every time-step since vehicle speed will be different for each time instant and dependant on the considered scenario. Nevertheless, it must be noted that even though those limits are presumably different for every time-step, they are assumed to be constant in the prediction module of the optimization algorithm. This approximation is a simplification, but helps maintaining the formulation in the linear domain.

Equation (23) can be alternatively represented in a more convenient representation for the optimization algorithm:

$$\begin{bmatrix} I \\ -I \end{bmatrix} U \leq \begin{bmatrix} U_{max} \\ -U_{min} \end{bmatrix} \tag{24}$$

Where *I* denotes the identity matrix.

2.5 MPC structure for Traffic Light problem

The methodology described until this point considered that the so called "crossing gaps" were already selected. For instance, if the vehicle was approaching a set of two Traffic Lights with three possible crossing gaps each, the green phases at which the vehicle would cross would be defined beforehand. In a real traffic scenario, it is reasonable to assume that several crossing gaps are available and feasible for the vehicle to cross at. Therefore, a

first preliminary assessment is required to distinguish which Gap configuration is more optimal and would output the lower energy consumption. The chosen approach in this Thesis is as follows:

A first quick assessment is performed and will determine which gap configuration is a priori most optimal. This is done through a MPC evaluation with long sample time, for each feasible TL gap configuration. Sampling time is longer in order to decrease calculation time since the number of cases to check rapidly increases with increasing number of TL. For instance, when considering 3 TL with 3 possible crossing gaps each, a maximum possible of 27 different combinations should be checked and compared among each other. For most of the cases, a large part of those combinations can be neglected since they represent cases which are not physically feasible.

In order to distinguish between different crossing gap combinations and to assess which would be most optimal, a criterion must be selected. The cost function was therefore integrated over the manoeuvre for each feasible configuration. The total additive costs for each combination differentiate the better alternatives. The gap combination giving the lowest cumulated cost function is therefore the a priori most suitable one. The selected crossing gap for each upcoming Traffic Light is then inputted to the short sample-time, high accuracy MPC.

Summarizing, the following steps are performed in the Fast MPC:

- A scenario with multiple Traffic Lights and multiple possible crossing gaps for each Traffic Light is initially considered.
- The problem is broken down to a sequence of multiple Traffic Lights with unique possible crossing gaps. At this stage, gap combinations which are not feasible are dismissed.
- Constraints are generated for each specific case as defined in Section 2.4.1.
- A quick MPC evaluation is performed for each individual combination to generate approximate vehicle trajectories, and cumulated cost function for this approximate trajectory is calculated.
- The process is repeated for all possible gap combinations. Cumulative cost functions for all combinations are compared among each other. The one with the lowest cumulative cost is chosen and forwarded to the high accuracy MPC.

The selected repetitive approach followed in high-level Fast MPC is described in Figure 8.

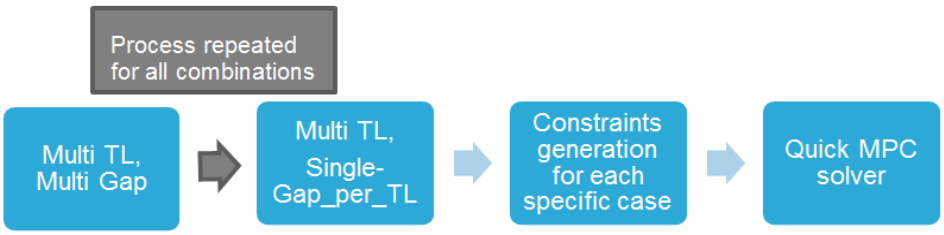

Figure 8: Gap selection process in high-level Fast MPC

There coexist therefore two parallel MPC algorithms as shown in Figure 9; the first one selects the most optimal gap to pass each Traffic Light (the so called Green Wave), whilst the second one is in charge of the precise control of the vehicle when using the optimal gaps from the other algorithm. The assessment done by the so called Fast MPC is only performed once at the beginning of each manoeuvre since this firstly calculated gap combination should remain the most optimal one as long as no unexpected events occur. As explained in the next section, the presence of other vehicles along the road is among the most important sources of disturbances.

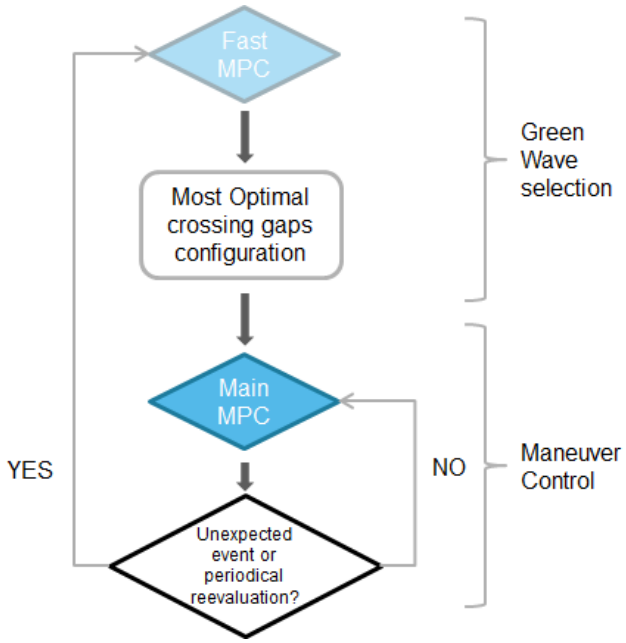

Figure 9: Fast MPC integration to ensure most optimal TL gap selection for all time instants

2.6 Traffic ahead consideration

Changing kinematics of vehicles along the road might restrict the access to some Traffic Light green phases. From a control perspective, it would be beneficial to predict behaviour of vehicles ahead of ego vehicle. To do so, it is assumed that the preceding vehicle's distance to ego-vehicle, speed and acceleration are obtained by measurements from on-board vehicle sensors (camera, radars, etc.).

An intuitive way to couple a preceding vehicle to the described problem approach is through introduction of additional position constraints to the problem formulation. Feasible region as defined in Figure 6 is then further restricted with information from other vehicles in the road (Figure 10). Note that only constant speed trajectory is depicted for simplicity. The estimated position of preceding vehicle can be described by basic equations of kinematics. Assuming constant acceleration of the preceding vehicle leads to its position being expressed as:

$$p_{pred,prec}(t) = p_{p0} + v_0 * t + \frac{1}{2} * a_{p0} * t^2 \tag{25}$$

After some testing, it was realized that acceleration/deceleration manoeuvres usually only last for a few seconds, and predictions beyond that time limit should be done without the acceleration term. The vehicle ahead is therefore assumed to follow its actual acceleration only for some time. After that time, the trajectory is assumed to be constant speed trajectory. As described below:

Preceding vehicle's trajectory: total prediction time: $t = t_{ca} + t_{cs}$ (26)

Constant acceleration for certain time t_{ca}

Constant speed after that time for a certain period t_{cs}

By appropriate tuning of those parameters, satisfactory realistic predictions of vehicle trajectories can be obtained. Again, the tuning of those parameters is a delicate process, and more accurate calibration techniques should be studied. A possibility would be to actively modify those time limits depending on actual vehicle speed, state of the traffic, etc. The reader is referred to Chapter 3, *Implementation challenges* for detailed argumentation regarding prediction parameters.

Once a suitable kinematic prediction model is obtained, coupling to current problem formulation is straightforward.

Predicted ego and preceding vehicle trajectories are linked through eq. (27) assuming that no overtaking manoeuvre is considered.

$$p_{pred,ego} \leq p_{pred,prec} - d_{safe} \tag{27}$$

d_{safe} indicates safety distance to preceding vehicle. For simplicity of implementation, this value is a user-defined parameter and is a constant value. For a more realistic implementation, this value could for instance, be a function of vehicle speed (as it is usually done in ACC functions, where it is proportional to speed with a small offset).

Eq. (27) is then combined with eq. (21). Upper and lower limits inputted to the optimization algorithm for predicted positions of the system are modified according to the predicted positions of vehicle ahead.

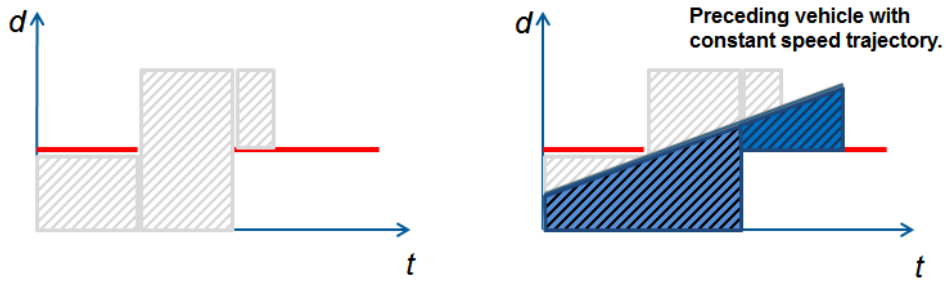

Figure 10: Feasible region modification due to preceding vehicle imposing new set of constraints.

In some cases, the behaviour of the preceding vehicle may impose deviation from the optimality, which was originally assessed via the previously mentioned Fast MPC approach. In other words, due to the vehicle in front, it might be impossible to cross TL at the desired gap. A possible scenario illustrating this concept could be a cut-in vehicle whose lower speed makes reaching the next TL in the desired green phase impossible. For such cases, the algorithm needs to launch a new gap evaluation, taking into account the recently detected preceding vehicle as seen in Figure 9. Since kinematics of preceding vehicle are a priori unpredictable for long term, the algorithm should periodically re-evaluate the most optimal gap combination based upon updated information. The frequency of recalculation ,f_{rec}, must be sufficient in order to capture all changing dynamics of the vehicle ahead, and is again a user-tuned parameter. Nevertheless, there must be a possibility for a non-scheduled recalculation due to the preceding vehicle significantly changing its kinematics. Additionally, another recalculation of the optimal gap is performed when a new Traffic Lights comes within the prediction horizon.

To summarize, three different events might trigger a Fast MPC evaluation:

- At the beginning of each manoeuvre, when the system is engaged.
- When a vehicle ahead is firstly detected and periodically after, with an update frequency.
- Preceding vehicle significantly changing its kinematics.
- New Traffic Lights come within the prediction horizon.

2.7 Complete Mathematical Problem Definition

This section summarizes collects and unifies the complete problem definition as explained in sections 2.1-2.6 under appropriate mathematical description:

The optimization problem solved at each time-step is to find the optimal control sequence *U(k)* such that it minimizes:

$$J_i(k_i) = \sum_{t=k_i}^{N_p-1} U^T R U + \left(y_{pred} - y_{ref}\right)^T Q \left(y_{pred} - y_{ref}\right)$$

$$S.T.$$

$$\begin{bmatrix} I \\ -I \\ N \\ -N \end{bmatrix} * U \leq \begin{bmatrix} U_{max} \\ -U_{min} \\ [x_{max}] - M * x(k_i) \\ -[x_{min}] + M * x(k_i) \end{bmatrix} \tag{28}$$

The two first lines in the above inequality refer to limits in acceleration and deceleration. The last two lines correspond to feasible positions as given by optimal gaps and preceding vehicle restrictions.

This formulation corresponds to a quadratic cost function in *U* (as demonstrated in Section 2.3) subject to linear inequality constraints in control, *U*. This problem formulation is well known and documented in the literature and efficient algorithms exist for its solution. MATLAB QUADPROG solver was used in this Thesis.

The optimization problem above is also complemented by the prediction model, stated as:

$$x_{pred} = M * x(k_i) + N * u_{opt,prev}$$

$$y_{pred} = C * x_{pred} = E * x(k_i) + F * u_{opt,prev}$$

Where,

$$x_{pred} = \begin{bmatrix} x(k_i+1) \\ x(k_i+2) \\ \dots \\ x(k_i+N_p) \end{bmatrix} \quad u_{opt,prev} = \begin{bmatrix} u(k_i+1) \\ u(k_i+2) \\ \dots \\ u(k_i+N_c) \end{bmatrix}$$

$$M = \begin{bmatrix} A \\ A^2 \\ \dots \\ A^{N_p} \end{bmatrix} \quad N = \begin{bmatrix} B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & \dots & A^{N_p-N_c}B \end{bmatrix}$$

$$E = \begin{bmatrix} CA \\ CA^2 \\ \dots \\ CA^{N_p} \end{bmatrix} \quad F = \begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

3 Implementation Challenges

The described methodology poses a large number of engineering challenges. First of them is to implement MPC formulation from scratch. A set of Matlab scripts dealing with Prediction and Optimization modules were created and their functionality was extensively tested to ensure correct behaviour. As a second stage, considerations regarding controller parameters tuning are presented.

3.1 Controller development in Matlab/ Simulink

The main challenge in this work was to manage Traffic Light scenarios and to create controllers from scratch based upon defined control strategies. The design process was always structured as follows: Controller design and functionality testing in Matlab Script environment. Once the desired functionality was obtained, the controller was taken to Simulink environment, and interfaced with CarMaker for Simulink models. The adaptation from script to CarMaker blocks was mainly handled through Embedded Matlab blocks. Such blocks allow for script-like programming and provide a straightforward adaptation. Nevertheless, some limitations appear:

- Embedded Matlab blocks use static memory allocation. This means that all internal variables need to be previously allocated in memory. No variable is allowed to change its size during execution time. This was initially a huge drawback, since many of the generated variables which deal with Traffic Lights are variable-sized. Matrices for prediction module for instance are changing their size dynamically depending on the actual prediction horizon. Nevertheless, some smart workarounds exist. Additional blocks were created that initialized size of matrices every iteration depending on current parameters. Those variables still are variable-sized, but their size is static for Embedded Matlab blocks.
- Solvers like MATLAB QUADPROG must be defined as extrinsic functions. This has implications for code generation. During simulation, the code generation software generates code for the call to an extrinsic function, but does not generate the function's internal code. Therefore, the simulation can only be run on platforms with installed MATLAB software. If a real implementation of controller in an independent environment is required, a custom QP solver algorithm should be developed.

An overview of controller structure as described in Section 2.5 can be found in Figure 11. The controller structure is as follows:

- Block 1: Raw Traffic Light data is processed in a block number 1. Input values are given correct format, and Traffic Lights are clustered in proximity. This is the first block in the processing sequence. Outputs are taken to blocks 2 and 3.

- Block 2: This block corresponds to the Fast MPC for efficient Green wave gap selection as described in Section 2.5. This block is a triggered subsystem, which only updates outputs under certain conditions. Signals produced in this subsystem, including the selected gap combination, are forwarded to main MPC controller.

- Block 3: The main MPC controller can be found in this block. Data from Fast MPC together with ego and preceding vehicle kinematics are imputed. The result is desired optimal vehicle acceleration.

- Blocks 4 and 5 can be found under Block 3. Block 4 is in charge of predictions of system behaviour based upon a model of the system. Generated variables are then employed in block 5, the QP solver. In this block, the constraints for the optimization problem are generated based upon updated information. Once the set of constraints and the cost function have been correctly generated, QP solver is called and the optimization problem is launched. Optimal control inputs are generated.

- Block 6 is a PI controller translating desired acceleration signals into corresponding gas and brake pedal signals.

- Block 7 defines needed variables for interfacing the controller with CarMaker and is also sending selected variables to Matlab workspace for results generation.

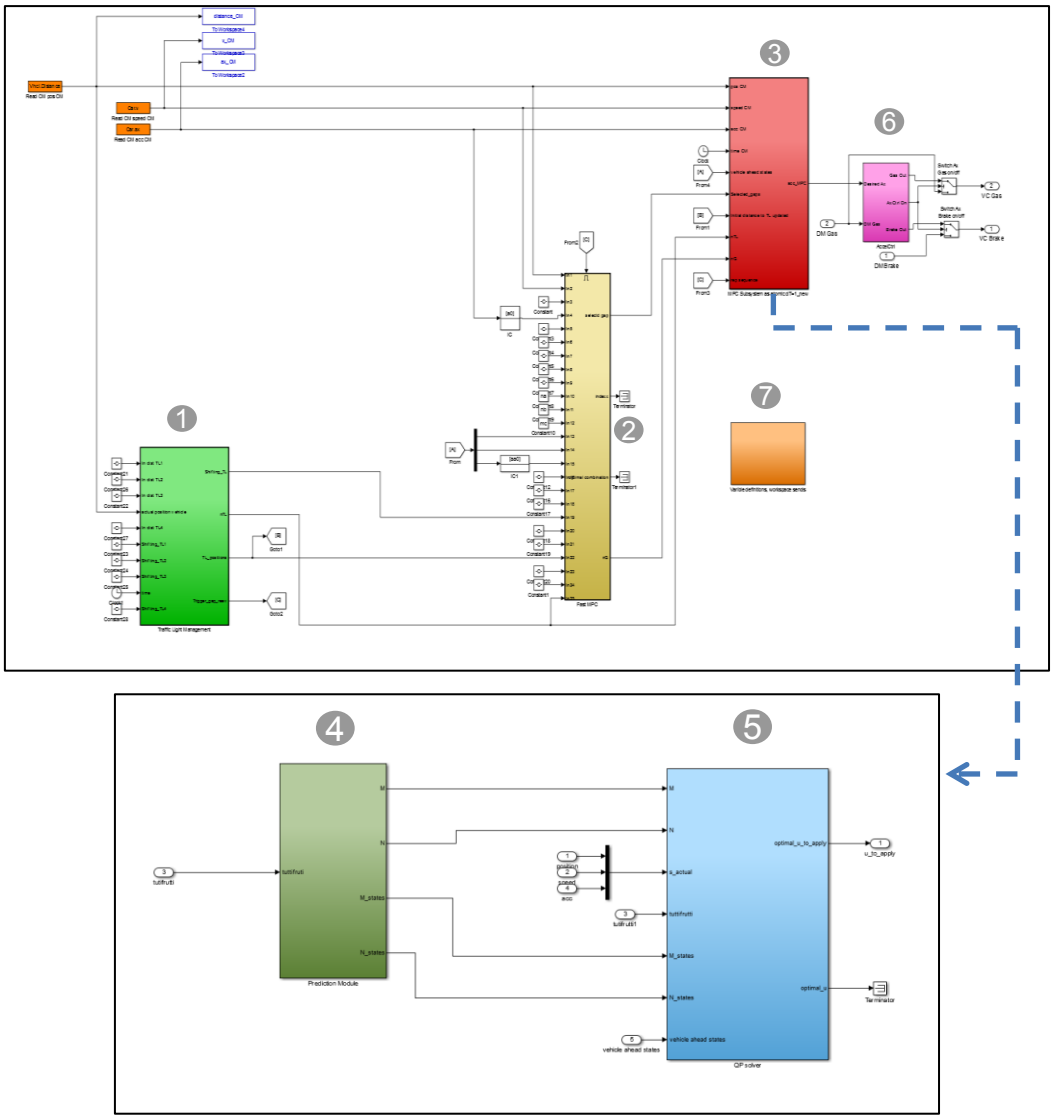

Figure 11: Simulink implementation of MPC controller

3.2 Pedal inputs corresponding to acceleration control

The developed controller was put together with the complex simulation environment of CarMaker. The generated controller outputted reference acceleration for the vehicle to follow. This reference acceleration needed to be transformed into throttle/brake signals via a PI controller (Figure 12). The PI controller structure was adapted from the ACC module already available, since this controller had reference acceleration as an input and was actuating throttle and brakes and was therefore already tuned for the application.

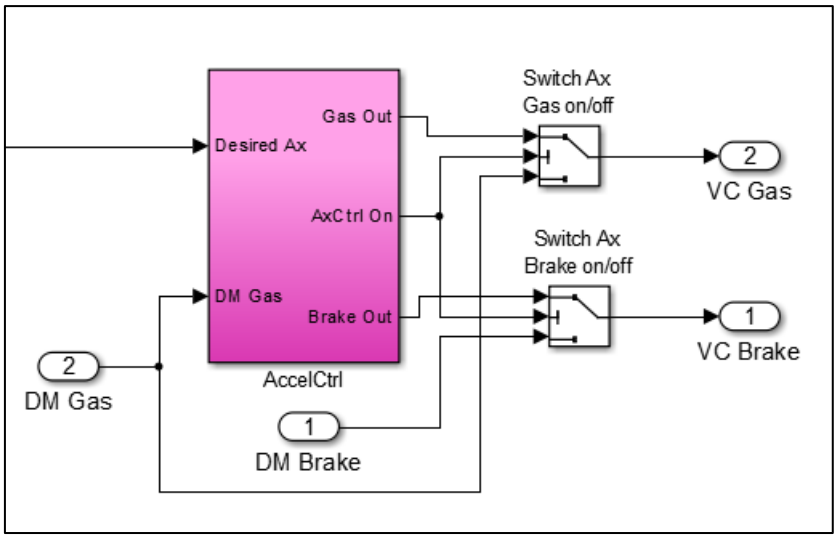

Figure 12: Detailed view of PI for Gas/Brake pedals from desired acceleration

3.3 Cost function relative weighting factors tuning

Another implementation challenge that requires dedicated effort is the tuning of values of Q and R weighting matrices for the cost function. Let's first recall the cost function:

$$J_i(k_i) = \sum_{t=k_i}^{N_p-1} U^T R U + \left(y_{pred} - y_{ref}\right)^T Q \left(y_{pred} - y_{ref}\right) \qquad (29)$$

The first term, weighted with matrix R, is trying to minimize acceleration. The second term, weighted by matrix Q, aims at minimizing differences between generated trajectories and reference trajectories. Since reference trajectories were selected to be position of the last Traffic Light, the second term is pushing for the vehicle to quickly arrive to TL position and consequently save travel time. Cost function is therefore trying to reach the last TL with lowest use of acceleration. Large values for weighting parameter R in relation to parameter Q will generate trajectories that present minimal deviation from initial vehicle path. If weighting balance is inverted, generated trajectories will tend to have high acceleration values to quickly reach TL position. In general, one would aim at energy-optimal trajectories, which mean that in practice, Q values are low in relation to R values

3.4 Relative tuning of Control and Prediction horizons

In general, Prediction Horizon, N_p is a fixed quantity for a specific scenario, and depends on how far in time are shifting phases of TL. Large predictions

horizons are beneficial for sake of stability of the closed loop behaviour as well for the global optimality of the solution. On the other hand, Control Horizon, N_c can be tuned. Experimental tests revealed that an increase in Control Horizon implied smoother solutions since the control action calculation considers larger time-frame in the horizon. If possible, large Control Horizons must be chosen. This implies increased matrix sizes for MPC controller, and therefore increased calculation time. Again, a trade-off arises. As a general rule of thumb, control horizons were kept in between 1/3 and 1/2 of Prediction Horizons.

Some tests were performed to illustrate the influence of increasing Control Horizon. The integration of cost function for a specific manoeuvre was used as metrics for improved solution.

Table 1: Values of cost function integration over time for different values of Control Horizons for a specific manoeuvre. (N_p=30 for all cases)

N_c value (s)	Value of the integration of J over time
7	81.40
15	10.16
30 (N_c=N_p)	8.94

Note how the case in which Prediction and Control horizons are equal is the one presenting lower integrated cost function value.

3.5 Tuning of revaluation frequency and vehicle ahead prediction horizon

Another challenge arises from selection of revaluation frequency, f_{rec} , and preceding vehicle prediction horizon. The first parameter indicates how often a Fast MPC evaluation is launched to check which gap configuration would give best performance. Second parameter defines how far in time predicted trajectory of preceding vehicle is projected. The larger this value, less probable becomes that real vehicle behaves as predicted.

Every certain number of iterations a fast MPC evaluation is launched to evaluate the "a priori" most optimal gap configuration. The reason behind this revaluation is that a vehicle ahead with an unpredictable trajectory could open new gap combinations or turn some combinations more optimal than the others.

Once an optimal configuration has been chosen, the system is forced to follow this gap combination until the new revaluation. In this period of time, the

system must satisfy the overall gap constraints together with the instantaneous preceding vehicle's positional constraints.

If too long vehicle ahead prediction horizons are chosen, the problem might very quickly become infeasible since the constant acceleration trajectories become too restrictive to cross any of the gap configurations (Figure 13), access to selected gap is prevented.

On the other hand, if the gap revaluations are not frequent enough, the vehicle in front might change its kinematics drastically in between two subsequent revaluations, and the problem might also become infeasible since the previously selected optimal gap configuration is not feasible anymore (Figure 14).

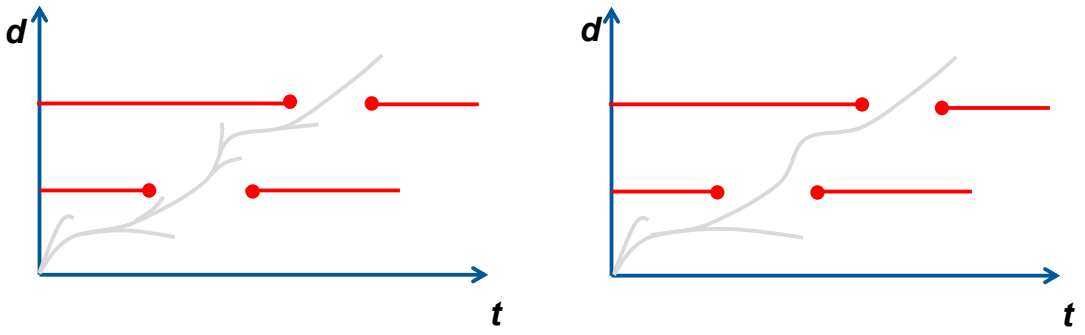

Figure 13: Problem turned infeasible due to too large vehicle ahead prediction horizons.

The curves in grey represent preceding vehicle's trajectory as well as predicted trajectories. It can be seen that for the right hand sided diagram, predicted trajectory is blocking the access to selected gap and therefore turning the problem infeasible.

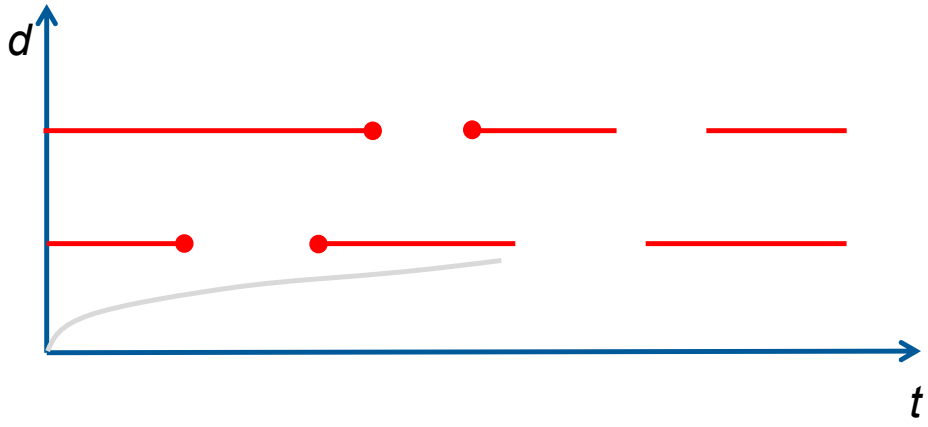

Figure 14: Problem turned infeasible due to not frequently enough gap revaluation.

It can be seen that the selected gap is not valid since preceding vehicle is following a slow trajectory. This scenario happens when revaluation frequency is not high enough.
The above descriptions exemplify the sensitivity in tuning of these parameters. In the tested code, revaluation frequency and vehicle ahead prediction horizon were chosen to be in the order of 5 seconds for both parameters. This allowed capturing kinematic variations in other vehicles for the simulated use cases. Nevertheless, as long as computational resources allow, revaluation frequency should be high. Vehicle ahead prediction horizon should be adapted to the type of driver. For instance, more aggressive drivers usually request higher acceleration and decelerations, and their prediction horizons should be shortened in order to avoid generating unfeasible trajectories. Summarizing, fine tuning of those parameters would require extensive testing and possibly some adaptive strategy which ensures stability under all possible disturbances.

4 Results and use-cases

This chapter will present the main results of testing of the Traffic Light Assistant system. It is essential that a consistent and clear testing methodology is established. The objective is to develop the metrics for energy consumption estimation. Since OpEneR prototypes are fully electric vehicles, a simple way to compare energy consumptions is by battery depletion. State of Charge (SoC) for the vehicles is compared through selected manoeuvres. In addition, the time required for the vehicle to finalize the manoeuvre (cross the last TL), is also defined as a comparison criterion. In general, the ideal controller would finalize the manoeuvre in the lowest time possible with lowest possible energy consumption. The baseline vehicle is the same OpEneR prototype but without access to TLA. The vehicle is controlled by IPGDriver module which is based on a statistical average driver. In order for comparisons between controllers to be fair, it is required that start and end positions and speeds are equal for both vehicles.

In order to present the results in an intuitive and didactic way, this section will introduce gradually increasing complexity in the selected scenarios: Firstly, a single Traffic Light will be considered. The amount of TLs will be gradually increased to finally end up with a real world road scenario with several clusters of TLs. In addition, a use-case in which a vehicle ahead is restricting the feasible solutions will also be presented.

Use-cases 1 to 3 will all be simulated with generic Traffic Light data. The purpose of such test cases is not to use realistic data, but to show the general controller's functionality and potential.

Selected use-cases aim at demonstrating the potential of TLA, but in order to assess complete controller performance, a whole statistical analysis should be performed.

Use-cases 1 to 3 are simulated with generic TL data shown in Table 2:

Table 2: Traffic Light data for use-cases 1 to 3

Traffic Light (if present)	**Absolute position along road [m]**	**Red phase duration [s]**	**Green phase duration [s]**
1	200	10	10
2	400	10	10
3	600	10	10

4.1 Use-case 1: Single Traffic Light

Ego vehicle initial kinematics can be found in Table 3.This firstly simulated use-case presents a road segment with a single TL in the horizon. As seen in Figure 15, normal unaware driver has to stop at the TL, while as MPC controlled vehicle smoothly accelerates enough just for the vehicle to cross TL under green to red switching. High energy savings can be observed in Figure 16. Normal vehicle is able to recuperate energy while braking for the TL, but there is huge energy consumption in accelerating back again to same end speed and route position as MPC controlled vehicle.

Initial ego vehicle quantities for both MPC and Normal driver are defined in Table 3.

Table 3: Initial kinematic quantities for use-case 1

	Initial speed [km/h]	**Initial position [m]**	**Initial acceleration [m/s²]**
Ego vehicle	32	0	0

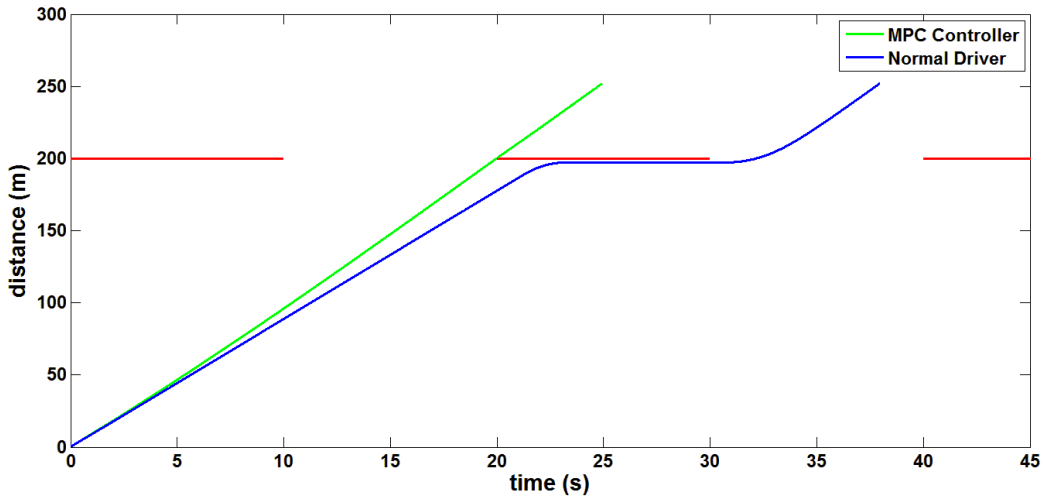

Figure 15: Trajectories for MPC and Normal Driver vehicles in use-case 1

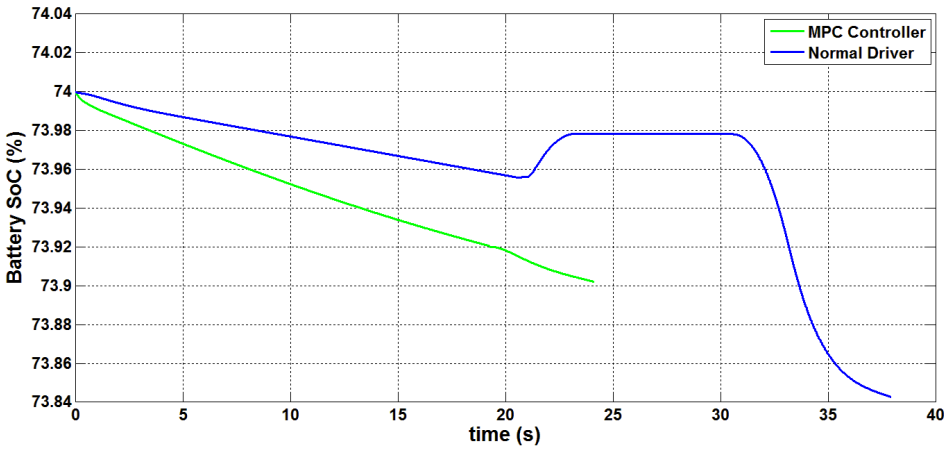

Figure 16: Battery SoC figure for MPC and Normal Driver vehicles in use-case 1

This use-case presents energy consumption reduction of 37.5% w.r.t. Normal driver and required travel time to complete the manoeuvre is 13.8 seconds lower, which accounts for a 36.4% reduction in total travel time.

4.2 Use-case 2: Two Traffic Lights

The second simulated use-case represents a road with 2 Traffic Lights with absolute positions and phase durations as defined in Table 4. Figure 17 depicts generated trajectories. MPC controller smoothly accelerates in identical manner to use-case 1 to avoid stopping at red. Afterwards, a constant speed trajectory is followed by MPC controller. This is an expected result, since cost function was mainly penalizing control action, vehicle acceleration in this case. Therefore, generated control inputs tend to zero when possible, leading to constant speed trajectories. In this case, Normal driver has to stop at the first TL but is able to cross the second one.

State of Charge, Figure 18, shows again significant energy savings, 31.6 % as compared to Normal driver. Time savings for this use-case account for 28%.

Table 4: Initial kinematic quantities for use-case 2

	Initial speed [km/h]	**Initial position [m]**	**Initial acceleration [m/s²]**
Ego vehicle	32	0	0

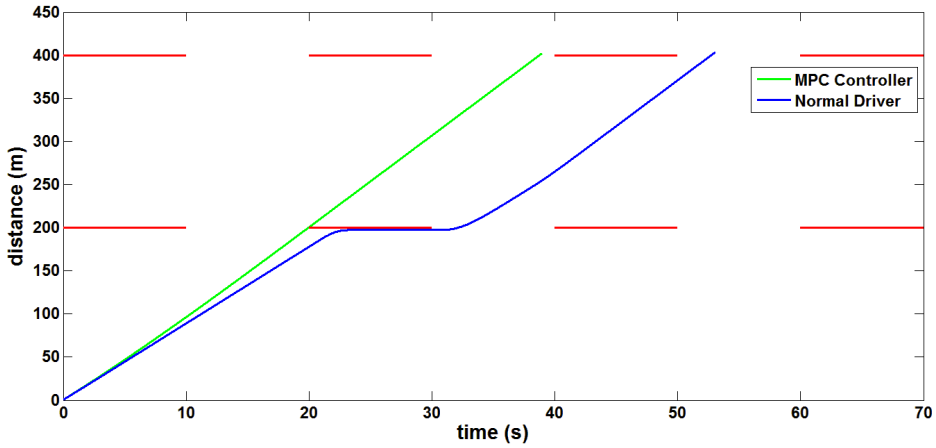

Figure 17: Trajectory plots for MPC and Normal Driver vehicles in use-case 2

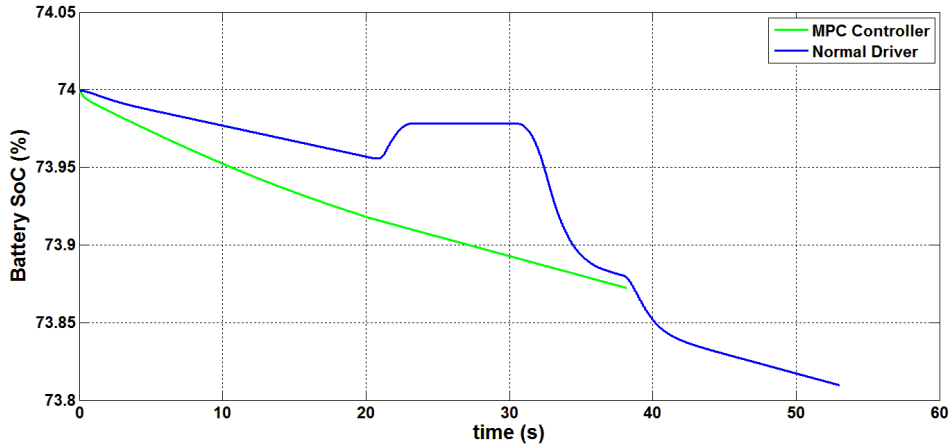

Figure 18: SoC evolution for MPC and Normal Driver vehicles in use-case 2

Note that normal driver SoC discontinuity is due to end speed adjustment reasons for fair comparison.

4.3 Use-case 3: Three Traffic Lights and Vehicle ahead

For this use-case, the number of Traffic Lights is increased to 3. This use-case includes a vehicle ahead along the road. Assuming that no overtaking is possible, that vehicle is restricting the access to most optimal gaps. This scenario illustrates the theoretical description under Section 2.6. The controller therefore needs to recalculate the most optimal gap configuration based on predictions of preceding vehicle's kinematics. Furthermore, use-case illustrates that generated logic is also able to cluster and optimize trajectories for up to 3 TL simultaneously. Test-case data can be found in Table 5. As seen in Figure 19, preceding vehicle follows a trajectory such that access to the second gap of second TL is blocked. MPC controlled vehicle "realizes" that the gap is inaccessible some seconds earlier, corresponding to vehicle ahead prediction

horizon. From that point on, a new optimal trajectory is generated to timely cross the last TL before it turns red.

Table 5: Initial kinematic quantities for use-case 3

	Initial speed [km/h]	**Initial position [m]**	**Initial acceleration [m/s²]**
Ego vehicle	28	0	0
Vehicle ahead	28	100	0

Regarding vehicle ahead prediction parameters:

-$Np, vehicle\ ahead = 7\ sec$, length of the prediction of the vehicle's ahead trajectory $t_{ca} = 2\ sec, \quad t_{cs} = 5\ sec$

-$dT, gaps\ selector = 4\ sec$, frequency of recalculation of the optimal gap combination.

For this use-case, Figure 20 shows SoC evolution for the manoeuvre. Again, MPC controlled vehicle achieves much better energetic results, reducing required energy by 36.8 %. On the other hand, for this scenario, manoeuvre completion time is marginally higher for MPC controlled vehicle, with an increase of travel time of 3.8% w.r.t. Normal driver.

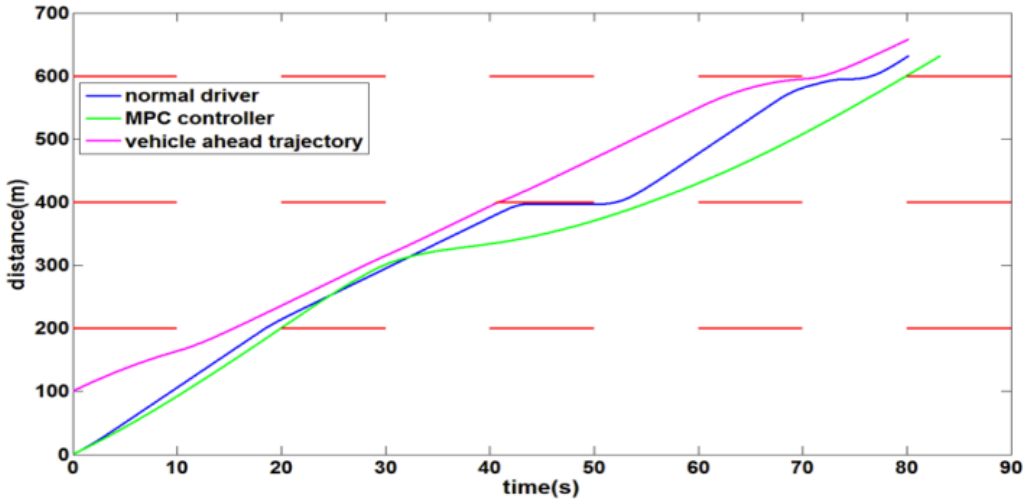

Figure 19: Trajectory plots for MPC, Normal Driver and preceding vehicle in use-case 3

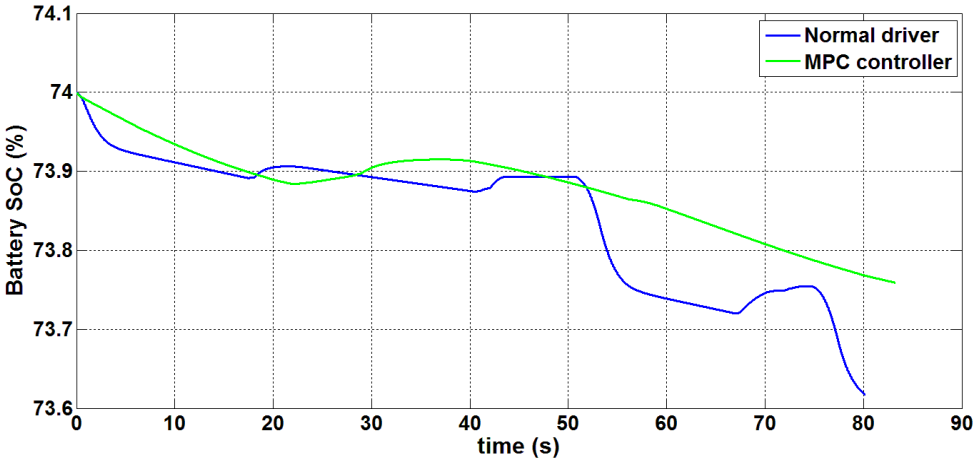

Figure 20: SoC evolution for MPC and Normal Driver in use-case 3

4.4 Use-case 4: Graz route segment with multiple Traffic Lights

This use-case represents a road segment in Graz containing a total of 14 Traffic Lights. Shifting information and absolute TL positions are obtained from OpenStreetMap [15] and are summarized in Table 6. Nevertheless, TL macroscopic control is a complex topic. Road authorities can actively adjust shifting depending on expected traffic flow, time of the day, and other factors. Therefore, obtained switching times represent an estimation of TL daily operating routines. Selected road segment starts at Keplerstraße, follows the Mur River along Lendkai, then along Brückenkopfgasse to close the circuit via Eggenberger Gürtel (Figure 21). Traffic Lights are distributed along the segment, and they are clustered based on proximity between lights. The MPC controller optimizes trajectories for each clustered group separately. For road segments in which no TL are in range, constant speed trajectories are assumed for both MPC and Normal driver controllers.

Table 6: Traffic Light data for selected route segment in Graz

Traffic Light number (clockwise)	Red Phase duration (s)	Green Phase duration (s)	Absolute Position (m)
1	10	30	150
2	10	30	380
3	10	20	880
4	20	10	1030
5	10	30	1300
6	26	20	2340
7	30	30	2530
8	30	30	2602
9	10	30	2902
10	10	30	3042
11	10	30	3172
12	10	30	3642
13	10	30	3835
14	10	30	4235

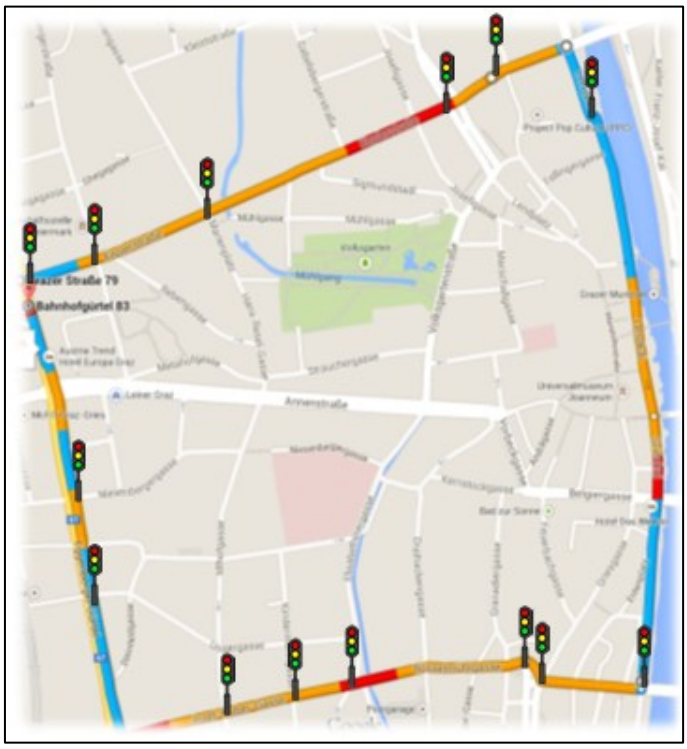

Figure 21: Google Maps representation of selected Graz road

As it can be seen in Figure 22, MPC controller successfully completes the road segment without stopping at any TL. Normal driver stops 4 times over the 14 TL. It is reminded that Traffic Light shifting is selected by road control organizations in such a way that vehicles are guided to catch the "Green Wave". Moreover, green phases are generally longer than red phases, which increases the possibilities for the unaware driver to cross under green light.

Figure 23 shows SoC evolution for both vehicles. It can be noticed how both states evolve in parallel and how energy is drained from the battery every time vehicle stops under red light. After 540 seconds manoeuvre, energy consumption of MPC equipped vehicle is reduced by 17%, while required completion time is 3.8 % lower. Lower values are obtained for this use-case as compared to previous scenarios, mainly due to a longer time frame and Normal driver not stopping so often.

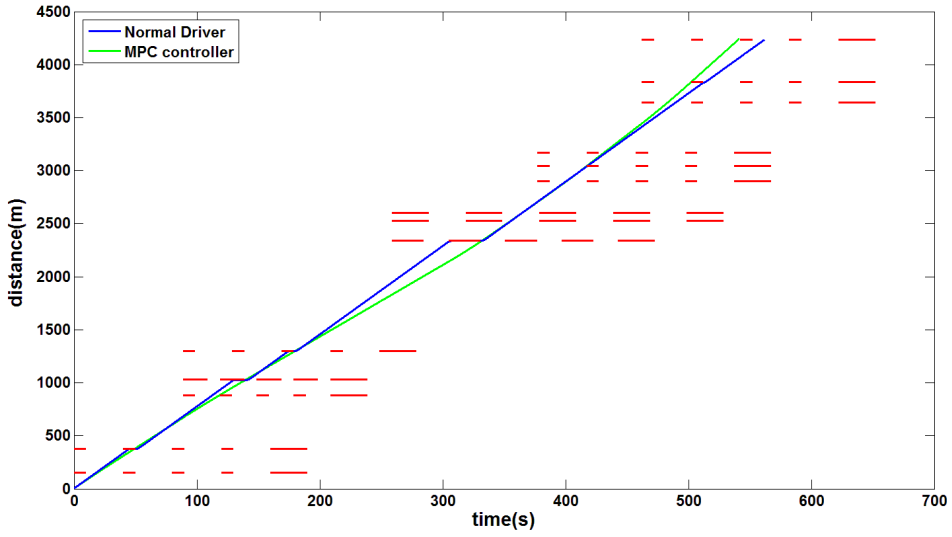

Figure 22: Trajectories for MPC and Normal Driver vehicles in Graz route

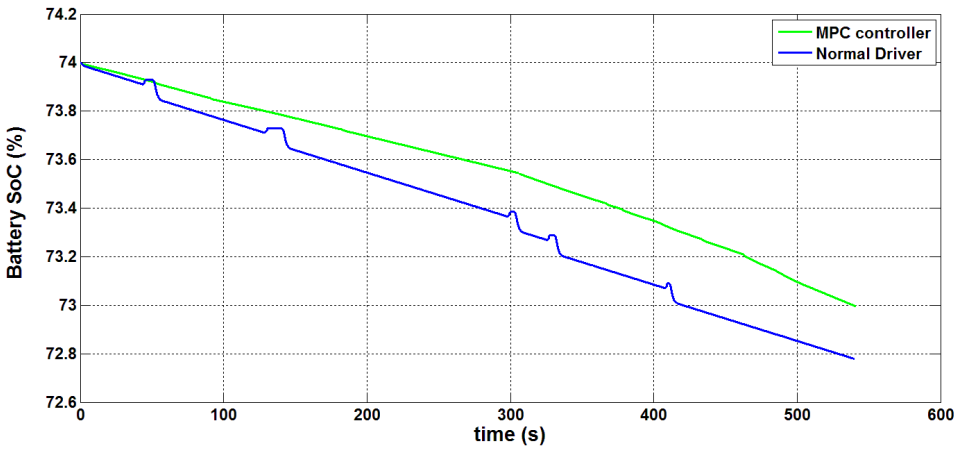

Figure 23: SoC evolution for MPC and Normal Driver vehicles in Graz route

Table 7: Energy and time savings for selected use-cases

Use-case	Energy saving (%)	Time saving (%)
1	37.5	36.4
2	31.6	28.0
3	36.8	-3.8
4	**17**	**3.8**

Note: negative time savings indicate an increase in required travel time w.r.t. normal driver.

Results for use-cases 1-3 indicate extremely large energy savings. These numbers must be taken with care: energy calculations are performed over a short time manoeuvre. If such energy consumption reductions are put into perspective, i.e. over a longer manoeuvre, the absolute energy consumptions takes more realistic values. Such behaviour can be seen in use-case 4. Even though short-time energy savings are extreme (especially if Normal Driver needs to stop at TL), long-time energy savings show a more realistic tendency.

This section demonstrated the energy and time saving potential of TLA. The Graz road segment demonstrated encouraging results, especially in energy savings. It was demonstrated that MPC approach is therefore a valid methodology for the approach of Traffic Light problem. It was also shown that the selected MPC methodology is able to efficiently deal with traffic ahead considerations, which brings the applicability of the controller closer to real world implementation. Additionally, the implementation of TL assistant on a relatively low portion of vehicle fleet might have positive effects on whole traffic fuel consumption, since the TLA equipped vehicles will probably force other vehicles to follow more energy efficient trajectories, resulting in multiplied energy savings. As it has been shown for ACC, even a low penetration rate has beneficial effects on fuel consumption for whole traffic.

5 Speed Advisory

The presented methodology has proven that MPC can be successfully applied to Traffic Light context with satisfactory results. Such advanced control technique has great potential and possible applications to any kind of powertrain. Nevertheless, it can be calculation intensive, especially if considered time horizons are large. VCU's capabilities would need to be enhanced with corresponding price increase. Therefore, next logic step was to simplify inherent algorithm complexity and to generate a "light but in-vehicle implementable" controller.

Back to the design phase, it was noticed that the problem structure could easily be condensed in matrix form. Number of Traffic Lights and number of gaps intuitively form a two dimensional matrix. Dynamic Programming-like techniques (DP) could possibly be applied. In DP, decisions are taken that minimize a certain cost. DP algorithms will examine all possible combinations and pick the best solution. [16].

The objective of this approach is therefore to present a very fast algorithm capable of recommending cruising speed for the driver to catch the green phases. In a similar way to Dynamic Programming algorithms, Speed Advisory is based on cost-to-go calculations. The energy required for the vehicle to cross a Traffic Light at a certain green phase is approximated by a cost function. Cumulative cost to cross the required TL's are calculated and the solution with lower cumulated cost is chosen as the most optimal one. Such a speed advisory system should not only provide driver with a recommended speed to cross the next TL in the closest green phase, but should take into account the multiplicity of TL and green phases, so that overall manoeuvre optimality is considered when providing a recommended speed. The critical point for such an approach is the selection of an appropriate cost function reflecting aforementioned criteria. The chosen cost function yields an approximation for the energy consumption required to go from gap *i* to gap *j*, see Figure 24:

$$W_{ij} = \theta_1 * \left(\frac{|v_0 * x_i - y_i + b|}{\sqrt{{v_0}^2 + 1}} + \frac{|v_0 * x_j - y_j + b|}{\sqrt{{v_0}^2 + 1}} \right) + \theta_2 * \Delta t \qquad (30)$$

$$W_{ij} = \theta_1 * f(deviation\ from\ traj.) + \theta_2 * f(time\ spent) \qquad (31)$$

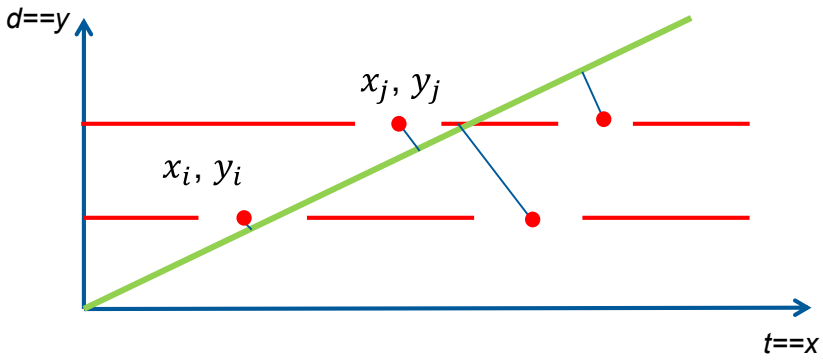

Figure 24: Schematic representation of deviation from initial trajectory criteria. Green line represents unaltered vehicle trajectory.

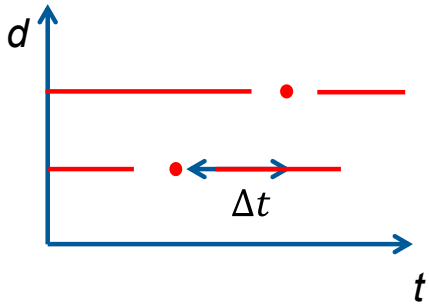

Figure 25: Delta time between two selected gaps.

As described in eq.(30,31), selected cost function is composed of two terms:

- A first term accounting for deviations of the selected gaps from the initial vehicle constant speed trajectory. (Figure 24). The idea behind this term is that gaps that initially are "closer" to the vehicle trajectory are most likely to be energy optimal, since no deviation from constant speed trajectory would be required. The main purpose is to alter vehicle's initial speed to a minimal extent.
- A second term accounting for time spent in between two consecutive green gaps. (Figure 25).When cumulative costs are calculated for the whole manoeuvre, this term accounts for total travel time. In this way, driver's desire to quickly cross the TL sequence is reflected.

The first term of selected cost function is the equation for the Euclidean distance from a straight line to a point. Straight lines take the generic equation in x-y coordinates:

$$a * x + b * y + c = 0 \tag{32}$$

Identifying terms with the equation of the unaltered constant speed trajectory, $= v_o * t + d_0$, it can be seen that the corresponding coefficients are $a = v_0$, $b = -1$, $c = d_0$.

Since the generic distance from a straight line as defined in eq. (32) and a point with coordinates (x_i, y_i) is :

$$d = \frac{|a * x_i + b * y_i + c|}{\sqrt{a^2 + b^2}} \tag{33}$$

By substitution of the previously calculated coefficients into eq.(33), the first terms of the cost function can be calculated. Coordinates for the points (x_i, y_i) are defined as the centres of the green gaps.

Balance between the two terms of the cost function is tuned through the weighting parameters θ. Again, selection of such parameters represents a challenge, since higher values of θ_2 in relation with θ_1 lead to gap selection aiming at a quick crossing of the TLs. If the parameters relation is inverted, the algorithm selects gaps which present low deviation from the vehicle initial trajectory. (Figure 26)

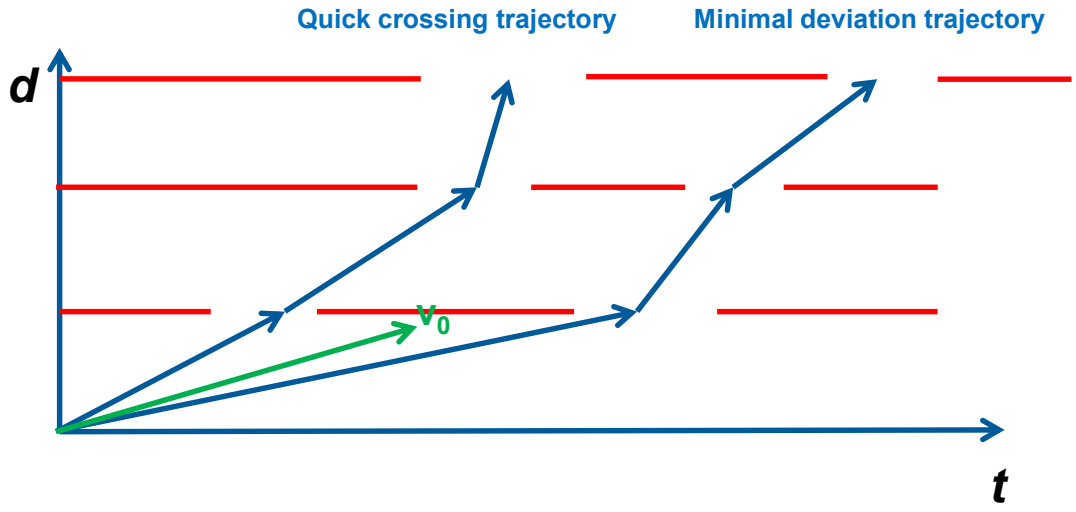

Figure 26: Quick crossing trajectories against minimal deviation from initial kinematics trajectory.

By observing Figure 27, it is clear that some gap combinations are not feasible For instance, going from 0 to 2 and then to 4 is impossible. In the same way, going from one gap to another one of same TL is not feasible. Cost-to-go for such combinations is set to infinity (Figure 28). As a consequence, only a few combinations are feasible since most of them have some infinite intermediate cost. This together with efficient coding makes calculation time very fast.

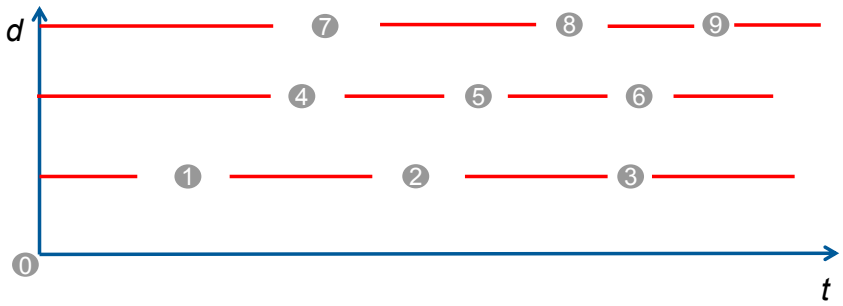

Figure 27: Exemplification of gaps available for a scenario with 3 TLs with 3 gaps each.

$$\begin{bmatrix} w_{01} & w_{02} & w_{03} & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & w_{14} & w_{15} & w_{16} & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & w_{25} & w_{26} & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & w_{36} & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & w_{47} & w_{48} & w_{49} \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & w_{58} & w_{59} \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & w_{69} \end{bmatrix}$$

Figure 28: Cost to go matrix representation. Rows represent starting gap, while columns represent end gap.

Once the most optimal gap combination is known, piecewise recommendation is given to the driver: required constant speed trajectories to cross each individual TL just in the middle of the green phase are calculated.

The described methodology presents high number of advantages, but also inherently some disadvantages:

- Contrarily to MPC formulation, the presented implementation does not take into account powertrain specific characteristics. This algorithm is therefore simple to implement on any vehicle.
- There is no consideration of any physical limitation in acceleration capabilities of the vehicle. This means that the controller can output too high acceleration values depending on the simulated case. Nevertheless, such disadvantage is easily compensated by adding logic that keeps the recommended speed within certain boundary.
- Speed Advisory is not able to deal with other vehicles in the road.

Despite those drawbacks, speed advisory performance was compared to MPC controller. It is expected that speed advisory would output worse results due to reduced complexity. A scenario where a vehicle is approaching a series of 3 Traffic Lights with equal regular shifting schedule is simulated. Energetic results for vehicle equipped with MPC, Speed Advisory (SA) system are presented in Figure 30. The baseline vehicle is labelled as Normal Driver, controlled by IPGDriver module and represents same vehicle without access to advanced information. It can be seen in Figure 29 that both MPC and SA are able to cross Traffic Lights without stopping at red while baseline vehicle

needs to stop twice. In addition, it can be seen how SA crosses TL just in the centre of the green gap.
From energetic perspective, State of Charge graph indicates that MPC controller provides, as expected, the best results. Speed advisory follows closely and demonstrates a good trade-off between complexity and energy consumption. Nevertheless, manoeuvre completion time is the highest for SA.

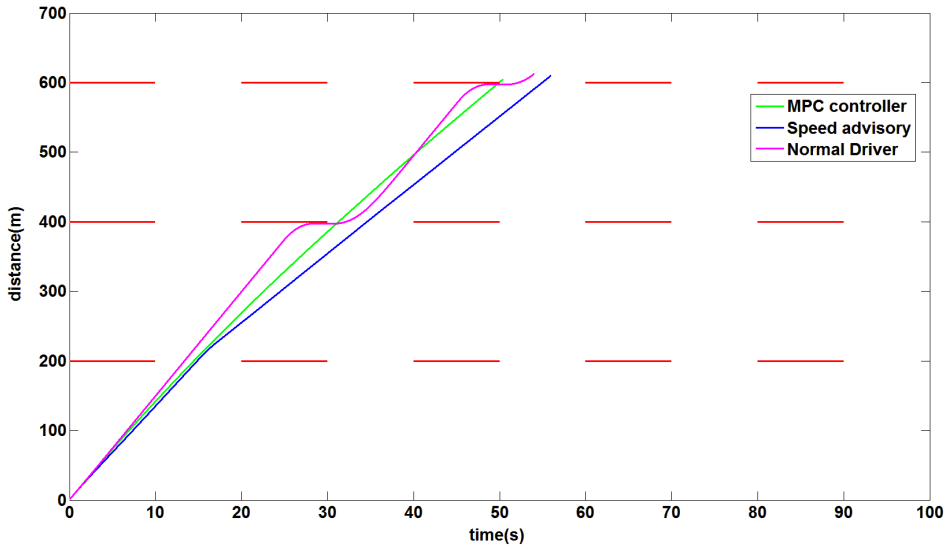

Figure 29: Trajectory plots for MPC, Speed Advisory and Normal driver vehicles

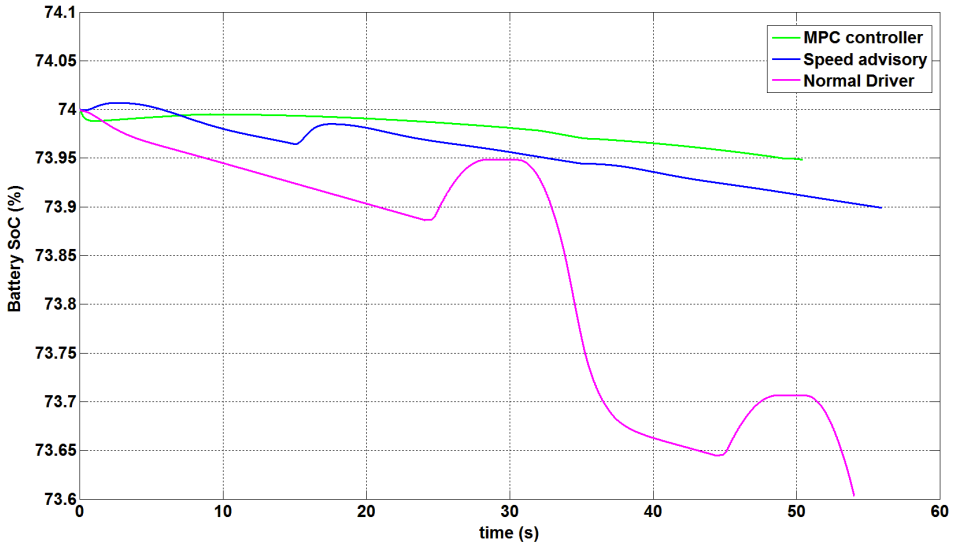

Figure 30: SoC plots for MPC, Speed Advisory and Normal Driver vehicles

5.1 Integration in Human Machine Interface software

A next step in the development process of advanced assistance functions is to include Human Driver in the Loop. This represents an important step towards more advanced and closer to real implementation functionalities. Such simulation

environment allows for functionality as well as specific software testing (Figure 32).

During the development of OpEneR project, advanced HMI software was developed by Centro Tecnológico de Automoción de Galicia (CTAG) for driver in the loop implementations. Such HMI is capable to display high number of indicators: actual speed, autonomy, blinkers, Electric Machine torques, recuperation capabilities, failures in multiple components… and more importantly upcoming traffic scenarios. The central console space is reserved for such scenarios management. Specific icons were developed for the implementation of TL assistance system. The HMI is capable of displaying actual state of Traffic Lights, recommended speed and distance to the next TL (Figure 31).

The physical OpEneR prototypes are equipped with such HMI software and displays and the communication is handled through CAN messages. Nevertheless, in office simulation context, the HMI software is emulated under Linux operating system and these CAN messages are created in a "CAN Bus emulator" in MATLAB/Simulink. The communication between HMI Software and the emulated CAN signals is handled via TCP/IP communications. The CAN messages are packed and sent over a specific IP address to the HMI software. Since HMI software was developed for Linux, emulation of such Operating System was required. For that purpose, a virtual machine was used which ran the required Linux platform (Figure 32).

In addition, driver interaction was allowed via Joystick input. Several buttons were directly linked to pedal and brake signals and the driver was able to, in real time, control the vehicle model based upon speed recommendation. As mentioned before, this layout allows for function development and HMI software testing.

Figure 31: Concept HMI for OpEneR prototypes developed by CTAG including custom icons for TL scenario management.

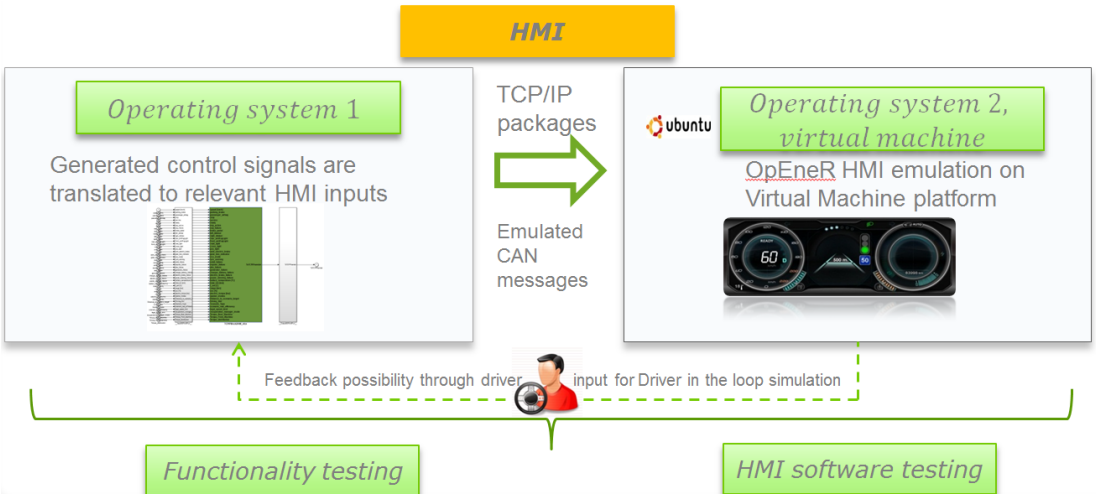

Figure 32: Layout for HMI testing using single simulation PC.

5.2 Integration in Mock-up

The previously described Speed Advisory will be tested in a simulator mock-up (Figure 34, Figure 35). To make that possible the developed function had to be adapted to the real time environment used in the simulator. This section describes the environment and the required Speed Advisory modifications.

The developed Speed Advisory System was adapted and integrated for Mock-up implementation in CTAG, Vigo, Spain as part of the OpEneR technologies demonstrator. All developed technologies were presented to open public and press. The mock-up available was equipped with Speed Advisory system displayed on HMI, and the driver had to actively control the vehicle to follow the recommendations and successfully crossing the Traffic Lights of a given test track (Figure 35).

This section will not present test results. As mentioned, the purpose was purely demonstrative. Nevertheless, the adaptation of the developed controller in real-time environment including a real driver in the loop was needed. The same OpEneR vehicle, powertrain and Energy Manager models were used and implemented in a Real Time environment with some interface modifications.

To do so, Real Time capable versions for each component of the tool-chain were used:

- Real Time capable CarMaker (InMotionTM)
- AVL Cruise RT
- Matlab/Simulink in Real Time Workshop

I/O communication was handled with AVL InMotion Real Time Platform. It allowed communication between simulator components (e.g. steering wheels and haptic pedals) and also to send suitable signals to HMI screen and

command haptic pedals. CAN Bus was used as the main communication bus with the same architecture as in the car. Test case definition and GUI animation was handled via a simulation PC (Figure 33).

Additionally, some modifications were necessary to redesign the interfaces between simulation tools and to fulfil real-time conditions. Main challenges in this respect regarded the adaptation of the designed controller to a Driver-in-the-Loop setup, enabling/disabling of the system and HMI signals handling. Driver acceptance was also assessed via pedal check. If the driver insistently kept on pressing the gas pedal, driver commands overrode the controller's signal and the control was given back to the driver. Summarizing, two different driving modes were distinguished, automated driving, in which the driver let the control to the vehicle, and manual driving, in which the driver had the control and choice to follow recommended speed. As mentioned before, any automation was subjected to driver acceptance. As soon as a cluster of Traffic Lights were detected in the vicinity, the system was engaged, and either speed recommendation or vehicle control actions were taken (Figure 39).

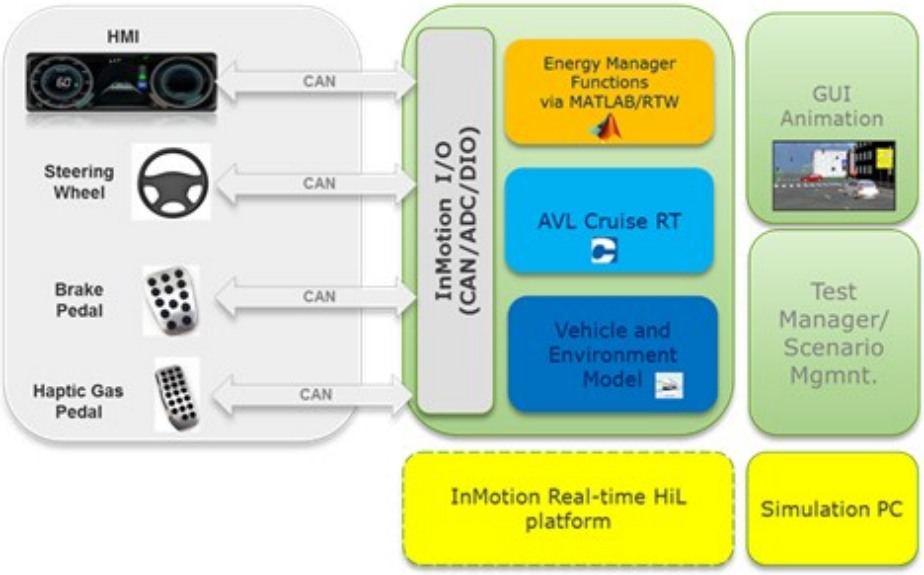

Figure 33: Structure for RT HiL Mock-up setup.

Figure 34: Detail of Mock-up HMI and HuD

Figure 35: Mock-up setup at CTAG

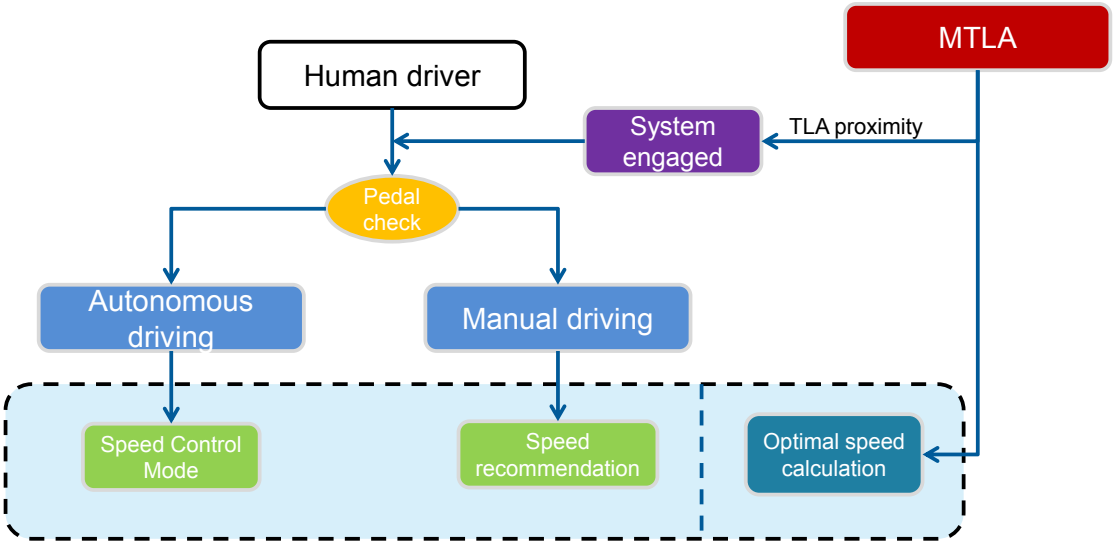

Figure 36: TLA interfaces to Driver

The functionality of Speed Advisory was successfully demonstrated in the described mock-up. Unfortunately, due to time limitations, the use-cases described in Chapter 4 were not reproduced in this new testing environment.

6 Parallel Investigations

On-board vehicle computation and storage capabilities are limited. Vehicles need nowadays to handle large amounts of data processing, and the optimization required by MPC approach is computationally demanding. This Chapter will briefly summarize different directions taken during this thesis project. As it was explained, MPC is relying on solving an online optimization at each iteration. The investigations described in this chapter aim at finding optimization-free alternatives to Model Predictive Control. In addition, performed investigations on Nonlinear MPC will be summarized.

6.1 Explicit MPC

This first subsection will describe the investigations performed regarding Explicit MPC (EMPC). Similarly to Speed Advisory, the idea was to develop optimization-free methodology which could be applied to Traffic Light scenarios. As it will be demonstrated, EMPC has great potential to be applied to any type of problems, but some limitations exist that made the approach to TL problems of no practical use. Nevertheless, results of conducted research are presented here.

Explicit MPC is a method for optimal control of processes with constraints where the control law is given in explicit form. Explicit MPC is derived in parallel to standard MPC, but the optimization problem is solved parametrically. Under certain assumptions of problem structure, it has been shown that the solution of the classical convex optimization problem takes the form of a piecewise affine function (PWA) defined over a polyhedral partition of the feasible system sate variables. The typical domain of interest is a subset of the state space, which is partitioned in a finite number of regions called critical regions. For each critical region, a particular state feedback control law yields the optimal control value.

Multi Parametric Optimization allows solving optimization problems as functions of certain number of parameters. Explicit MPC solves the optimization problem with system states as the parameters of the system. By doing so, the optimal control input can be evaluated on-line via a simple function evaluation. The computational effort is condensed off-line, where the set of optimal controls is calculated as functions of partitions of the feasible states of the system.

In the implementation there is no need for repetitive optimization since only the explicit solution is evaluated every sampling instant with reduced computational effort as demonstrated by *Herceg and Scibilia* [17,18].

The quadratic programming problem defined in 2.7 can be easily adapted to a Multi Parametric Quadratic Program (MPQP) through the change of variable as

demonstrated by *Francesco Scibilia* in [17]. By applying KKT conditions to the MPQP, an explicit expression for the optimal solution as function of the parameter x, state space, for each critical region can be obtained.

Due to time limitations, the MPT3 optimization toolbox was used. This toolbox is an open source, Matlab-based toolbox for parametric optimization, computational geometry and model predictive control developed by the ETH Zürich [19]. YALMIP is used in MPT3 background as a modelling language for advanced modelling and solution of convex and nonconvex optimization problems.

One of the main advantages of using MPT3 toolbox is the ease of implementation of standard MPC formulations. Plant model, cost function and simple constraints are easily implemented and the problem as defined in chapter 2 could be reproduced. Nevertheless, custom constraint definitions were required to match the complex system of constraints needed for Traffic Light management as defined in section 2.4. The underlying modelling code in YALMIP was therefore adapted and modified.

Once the MPC controller is fully defined, the toolbox allows for explicit MPC code generation. The result is a simple Matlab function outputting the optimal control input as function of the system states. Figure 37 describes the regions of the polyhedral partition of the state space, and Figure 38 depicts the optimal control input for each different Critical Region for specific simulated use-case Note that only the first element of the optimal control sequence is depicted.

The simulated use case was:

Table 8: Traffic Light data for selected manoeuvre

Traffic Light	Position (m)	Green gap (s)
1	100	5-10
2	200	25-20

In addition, constraints on system acceleration were added as maximum and lower limit of [5,-5] m/s^2 respectively.

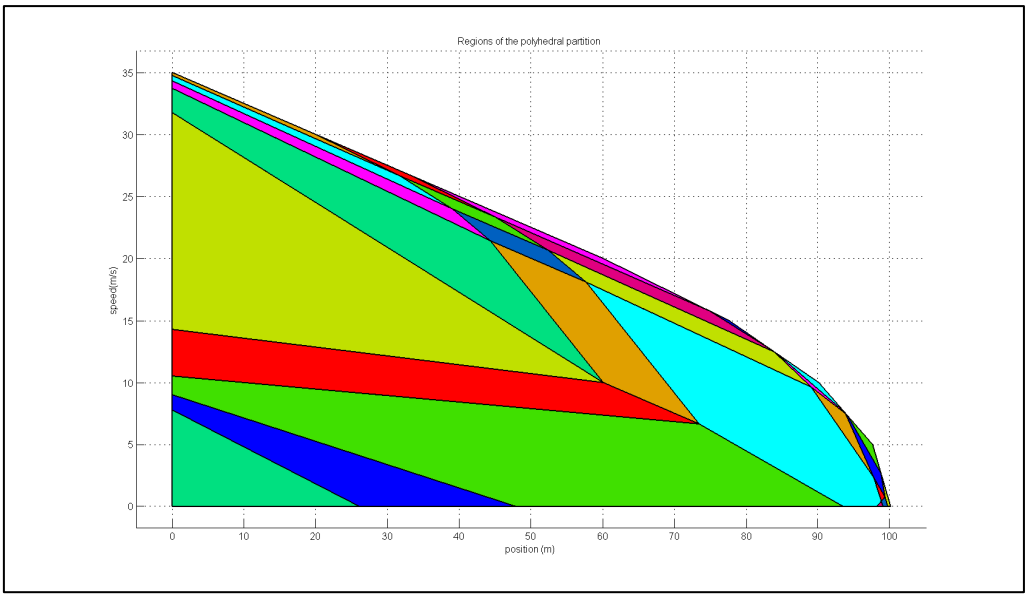

Figure 37: Polyhedral State Space partition in Critical Regions.

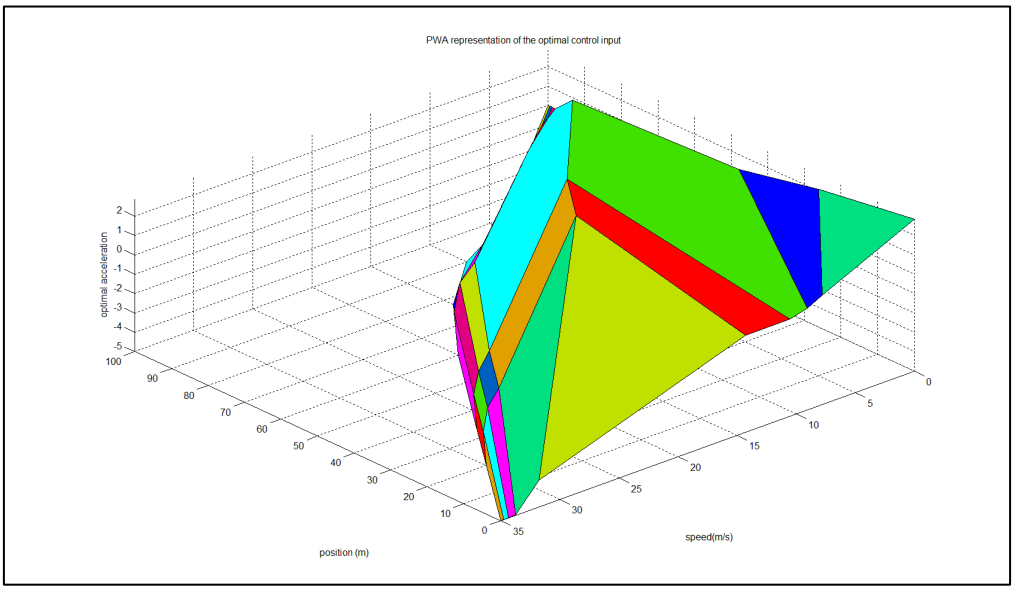

Figure 38: Optimal control inputs for each different Critical Region.

Despite the promising expectations of this method, it was finally concluded that EMPC could not be applied to the Traffic Lights problem: the proposed MPC approach is relying on dynamically changing constraints. As soon as the vehicle moves from one time instant to the next one, the whole set of position constraints is shifted a time unit to emulate the TL green gap approaching in time. This is a key feature of the proposed controller but makes explicit approach impracticable since one should generate a different explicit controller for each specific scenario, which is of no practical use. Nevertheless, it must be highlighted that this methodology has a great potential for other formulations.

6.2 Non-Linear MPC

In general, NMPC problem is formulated similarly to MPC: solving an on-line open-loop optimal control problem with a finite horizon. Such problem is completed with constraints involving system states and controls. A nonlinear prediction model is defined as:

$$\dot{x} = f\big(x(t), u(t)\big) \quad x(0) = x_0 \tag{34}$$

$$y = g(x(t), u(t)) \tag{35}$$

where f, g are nonlinear functions.

Furthermore, the constraints are of the shape:

$$u(t) \in U, \forall t \geq 0 \quad x(t) \in X, \forall t \geq 0 \tag{36}$$

Here the set of feasible input and states values are denoted by U and X respectively.

Nonlinear MPC is therefore characterized by the use of nonlinear system models in the prediction module. While generated optimization problems are convex in MPC, they are not convex anymore in NMPC. This poses challenges for stability and solution of such problems. In addition, the complexity of the optimization problem translates into an important increase in computation time. Furthermore, obtaining nonlinear models for some processes is a challenge by itself. Several techniques exist such Neural Networks, Local model Networks, empirical models, etc. [6]

The following key characteristics define NMPC:

- NMPC allows use of nonlinear prediction model
- Similarly to MPC, NMPC allows explicit consideration of constraints on inputs and states.
- In MPC and NMPC, a certain performance criteria shaped under a cost function is minimized on-line.
- Predicted system behaviour and closed loop behaviour differ in general. [20].

It can be concluded that NMPC allows for accurate representation of non-linear systems. Most of real world processes are inherently non-linear. Being able to represent such dynamics into the system model opens the gate to advanced control strategies. For instance, explicit vehicle dynamics, and electric machine efficiency could be taken into account directly in the system model.

The main drawback of NMPC is the difficulty to solve constrained nonlinear optimization problems. Methods to solve such problems cannot guarantee global optimality of the solution.

The following section describes in a simplified manner different alternatives available to deal with nonlinearities (Figure 39):

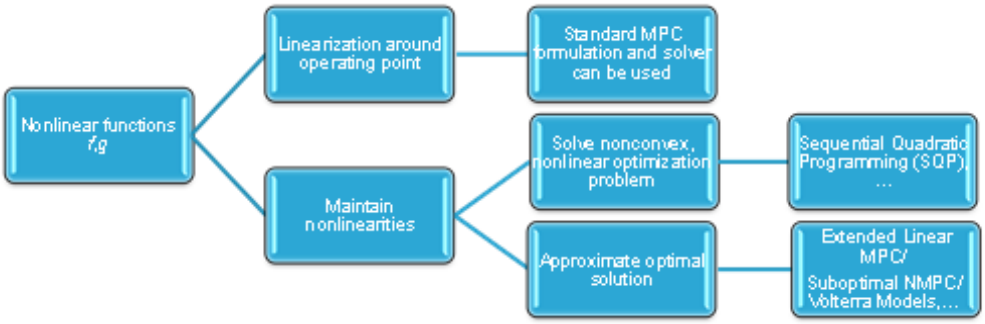

Figure 39: Schematic for possibilities to deal with nonlinear MPC

The first method to deal with nonlinearities is to linearize the system equations around an operating point. With this approach, the standard MPC formulation can still be used. The only difference is that plant model consists of a simplified version of complex plant dynamics. Nevertheless, if operating point is too far away from the linearization point, huge errors might occur. Therefore, a constant relinearization is needed to guarantee acceptable operating conditions.

On the other hand, if nonlinearities are preserved, two alternatives are available:

- The nonlinear, nonconvex optimization problem can be directly solved. This is usually done using Sequential Quadratic Programing (SQP). SQP is an iterative technique in which a sequence of quadratic problems subject to linear approximations of the nonlinear constraints. Nonlinear Optimization Problem is successively replaced by a problem in which the objective is replaced by a quadratic approximation. The exact solution of such problems is a difficult task and there is no guaranteed global optimality.
- An alternative to directly solving the nonconvex optimization problem is to approximate the optimal solution. Efficient formulations have appeared in recent years. All of them try to avoid directly solving the nonconvex optimization problem:

    - Extended Linear MPC: this is one of the simplest methods. The idea is to add a new term to the prediction module that tries to take into account nonlinearities in the process and compensate for disparities between nonlinear and linear models.

    - Suboptimal NMPC: The underlying principle is to stop optimization when satisfying enough results have been obtained and stability can still be guaranteed. It can be shown that under certain circumstances, the solution is guided towards a continuous decrease in cost. The main technique that uses this concept was proposed by *Scokaert, Mayne, and Rawlings* [21].

- MPC based on Volterra Models: In this method, it is required that the Nonlinear Problem has a special structure, with polynomial nonlinearities in this case. The nonlinear problem is solved by iteration of the linear solution, based on that particular structure. This can be exploited to achieve feasible solutions for the general optimization problem.

7 Conclusions and Future Work

The developed methodology has proven huge potential regarding energy and time savings. Energy savings between 17 and 37 % were observed depending on the selected use-case. Time savings present in general a reduction tendency, again subjected to use-case. It has been demonstrated that MPC is an applicable methodology for handling of the TL problem: TL crossing gaps can be efficiently represented through constraints and the selected minimization criteria represents a good approximation of energy consumption. Furthermore, presented MPC methodology is able to deal with a preceding vehicle in the road and to keep energy-optimal trajectories.

Model Predictive Control is relying on solving an optimization problem every iteration. Therefore, its computational load is not negligible. As a consequence, an optimization-free algorithm was developed. Speed Advisory uses a cost-to-go approximation of energy consumption to select the most optimal crossing gaps. Human-in-the-Loop testing of the Speed Advisory was implemented via HMI software which was part of OpEneR project. Furthermore, the developed controller was adapted for implementation in a Mock-up simulator at CTAG, for OpEneR final review meeting.

Finally, one should remember that there is the potential to apply MPC approach to any kind of powertrain topology. This opens up the possibility for more detailed efficiency considerations, resulting in better control. Nevertheless, further investigation on Nonlinear MPC and especially on methods for solving such complex non-convex optimization problems is required.

During development of this thesis, several possible work extensions or topics where further work would be of interest were found:

- Extend the MPC formulation to a more precise powertrain description, enabling enhanced control strategies. On the other hand, full powertrain efficiency representation might lead to non-linear control strategies. However, it should be emphasized that the Traffic Light problem formulation is valid independently of the specific powertrain.
- Study a better performing algorithm for the Fast MPC approach. Instead of checking all possible gap combinations at once, the number of gaps considered as well as the combinations should be checked progressively. One way of achieving this could be to launch a first optimization round for the gap combinations which would probably the most energy efficient, and to progressively extend the calculations to adjacent gaps until no improvement in cost function is found.
- Develop and efficient clustering TL algorithm. Perform some literature research to define the optimal size of TL clusters: e.g. a group of 4 incoming TL should be clustered as 2+2 or 4 at once and which are the benefits/drawbacks of large clusters. Larger clusters inherently require

higher computations. The trade-off between computations and energetic gain should be analysed.

- Traffic behind consideration. Consider vehicles behind in the problem formulation with the aim of letting as many vehicles as possible pass a Traffic Light.

8 References

[1] European Association for Battery electric Vehicles. “Energy consumption, CO2 emissions and other considerations related to Battery Electric Vehicles” 2009.

[2] U.S. Department of Transportation.Federal Highway Administration. “Signal Timing Manual”. 2008.

[3] http://www.fp7-opener.eu/

[4] D. Hrovat, S. Di Cairano, H.E. Tseng, I.V. Kolmanovsky: “The Development of Model Predictive Control in Automotive Industry: A Survey”. 2012 IEEE International Conference on Control Applications, 2012 Dubrovnik, Croatia.

[5] J.M. Maciejowski: “Predictive Control with Constraints”, Prentice Hall, 2000.

[6] Edurador F. Camacho, Carlos Bordon: “Model Predictive Control”, Second Edition. Advanced Textbooks in Control and Signal Processing. . Springer, 2007

[7] Liuping Wang, “Model Predictive Control System Design and Implementation using MATLAB”. Advances in Industrial Control, Springer, 2008.

[8] R.Fischer, K.Küpper, S.Jones, E.Kural. “The connected powertrain”. 25th International AVL Conference “Engine and Environment”, September 2013, Graz, Austria.

[9] S.Jones, E.Kural, K.Knödler, J.Steinmann, V.Braun. “Simulated development of safe and energy efficient driving of the 4WD OpEneR electric vehicle with an advanced cooperative regenerative braking system”. 9th ITS European Congress, Dublin, 2013.

[10] S.Jones, A.Huss, E.Kural, R.Albrecht, A.Massoner, K.Knödler. “Optimal Electric Vehicle Energy Efficiency and Recovery in an Intelligent Transportation System”. 19th ITS World Congress, Vienna, 2012.

[11] S.Jones, A.Huss, E.Kural, A.Massoner, S.Ludewig, K.Knödler, J.Steinmann, S.Laversanne. “Seamless development of vehicle energy management, recuperation and safety systems: Pure office simulation to 4WD powertrain testbed”.

[12] Emmanouil Koukoumidis, Li-Shiuan Peh, Margaret Martonosi: “ SignalGuru: Leveraging Mobile phones for Collaborative Traffic Signal Schedule Advisory” .

[13] Behrang Asadi and Ardalan Vahidi: Predictive Cruise Control: “Utilizing Upcoming Traffic Signal Information for Improving Fuel Economy and Reducing Trip Time”. IEEE Transactions on Control Systems Technology, vol. 19, No.3, May 2011.

[14] The MathWorks Inc., "Matlab R2013a Documentation".

[15] www.openstreetmaps.com

[16] Bellman R., “The Theory of Dynamic Programming, The RAND”. 1954.

[17] Francesco Scibilia: “Explicit Model Predictive Control: Solutions Via Computational Geometry”, Doctoral Thesis at NTNU 2011.

[18] Martin Herceg, “Real-Time Explicit Model Predictive Control of Processes”. Dissertation Thesis, Slovak University of Technology, 2009.

[19] http://control.ee.ethz.ch/~mpt/3/Main/HomePage

[20] Rolf Findeisen, Frank Allgöwer, “An Introduction to Nonlinear Model Predictive Control”.

[21] P.O.M. Scokaert, D.Q. Mayne, and J.B. Rawlings: “Suboptimal model predictive control (feasibility implies stability)”. IEEE Transactions on Automatic Control, 44(3):648–654, 1999.

[22] E.Kural, B.A. Güvenç, “Model Predictive Adaptive Cruise Control”.

[23] Weingerl, Patrick: “Optimal Energy Management System for an Electric Vehicle in an Intelligent Transportation System”, Master Thesis.

[24] Massoner Alexander: “Optimum Energy Management Strategies for an Electric Vehicle Integrated in an Intelligent Transport System”. Master Thesis.