Cooperative ADAS and Driving, Bio-Inspired and Optimal Solutions

Submitted as Research Report / Honours / Master Dissertation in SIT723/SIT724

Giammarco Valenti

Supervised by: Prof. Francesco Biral, Prof. Mauro Da Lio

Contents

List of Figures

List of Tables

Dedicato a Cecilia e alla mia famiglia, per avermi supportato e per aver creduto in me.

Abstract

Mobility is a topic of great interest in research and engineering since critical aspects such as safety, traffic efficiency, and environmental sustainability still represent wide open challenges for researchers and engineers. In this thesis, at first, we address the cooperative driving *safety* problem both from a centralized and decentralized perspective. Then we address the problem of optimal energy management of hybrid vehicles to improve environmental *sustainability*, and finally, we develop an intersection management systems for Connected Autonomous Vehicle to maximize the traffic *efficiency* at an intersection. To address the first two topics, we define a common framework. Both the cooperative safety and the energy management for Hybrid Electric Vehicle requires to model the driver behavior. In the first case, we are interested in evaluating the safety of the driver's intentions, while in the second case, we are interested in predicting the future velocity profile to optimize energy management in a fixed time horizon. The framework is the Co-Driver, which is, in short, a bio-inspired agent able both to model and to imitate a human driver. It is based on a layered control structure based on the generation of atomic human-like longitudinal maneuvers that compete with each other like affordances. To address driving safety, the Co-Driver behaves like a safe driver, and its behavior is compared to the actual driver to understand if he/she is acting safely and providing warnings if not. In the energy management problem, the Co-Driver aims at imitating the driver to predict the future velocity. The Co-Driver generates a set of possible maneuvers and selects one of them, imitating the action selection process of the driver.
At first, we address the problem of *safety* by developing and investigating a framework for Advanced Driving Assistance Systems (ADAS) built on the Co-Driver. We developed and investigated this framework in an innovative context of new intelligent road infrastructure, where vehicles and roads communicate. The infrastructure that allows the roads to interact with vehicles and the environment is the topic of a research project called SAFESTRIP. This project is about deploying innovative sensors and communication devices on the road that communicate with all vehicles. Including vehicles that are equipped with Vehicle-To-Everything (V2X) technology and vehicles that are not, using an interface (HMI) on smart-phones.
Co-Driver-based ADAS systems exploit connections between vehicles and (smart) roads provided by SAFESTRIP to cover several safety-critical use cases: pedestrian protection, wrong-way vehicles on-ramps, work-zones on roads and intersections. The ADAS provide personalized warning messages that account for the adaptive driver behavior to maximize the acceptance of the system.
The ability of the framework to predict human drivers' intention is exploited in a second application to improve *environmental sustainability*. We employ it to feed with the estimated speed profile a novel online Model Predictive Control (MPC) approach for Hybrid Electric Vehicles, introducing a state-of-the-art electrochemical model of the battery. Such control aims at preserving battery life and fuel consumption through equivalent costs. We validated the approach with actual driving data used to simulate vehicles and the power-train dynamics.
At last, we address the traffic *efficiency* problem in the context of autonomous vehicles crossing an intersection. We propose an intersection management system for Connected Autonomous Vehicles based on a bi-level optimization framework. The motion planning of the vehicle is provided by a simplified optimal control problem, while we formulate the intersection management problem (in terms of order and timing) as a Mixed Integer Non-Linear Programming. The latter approximates a linear problem with a powerful piecewise linearization technique. Therefore, thanks to this technique, we can bound the error and employ commercial solvers to solve the problem (fast enough). Finally, this framework is validated in simulation and compared with the "Fist-Arrived First-Served" approach to show the impact of the proposed algorithm.

0.1 Introduction and motivation

Road traffic accidents are still one of the leading causes of death and injuries worldwide. According to the most recent World Health Organization report, [98], 1.3 million people die every year from a road accident, and between 20 and 50 million people suffer non-fatal injuries.
In the early years of this century, we can see a downward trend in the number of accidents and injuries severity. Unfortunately, this encouraging trend has slowed down and reached a plateau [96][1][4]. For vulnerable road users (acrshortvru), the trend is even in the opposite direction, with a rise in fatalities. Almost 40% of fatalities regards pedestrians in urban areas. This portion adds up to 12% of cyclists' fatalities and 18% of PTW's, reaching 70% of total fatalities in urban areas for these categories of users. Outside of urban areas, this percentage is still high (32%). In Europe, critical infrastructure situations are responsible for 35% of the road accidents [2] mainly

for low visibility and unexpected road surface conditions. At junctions, 20% of total accidents occur, weather conditions or impaired visibility cause many of them.
The increase in road networks and related traffic is frequent temporary maintenance interventions with roadworks that impact road safety and traffic efficiency. The beginning of work zones is the most critical point for such areas, especially if one or more are close. Inattention mainly causes rear-end accidents [12][11]. Additionally, work zones represent a danger for road-workers that may be hit by oncoming vehicles as reported in[2]. Furthermore, roadworks strongly affect traffic efficiency. The layout of the work zones determines how the traffic is affected [12].
The current status of the ADASs provides very effective safety systems within the limit of the onboard sensors. For instance, the emergency braking system effectively avoids car to pedestrian crashes and rear-end crashes. However, the still low penetration rate of such systems and the limits of their capabilities, together with those above not encouraging numbers for accidents, motivates the development of new technologies that aim at overcoming the current safety systems limits. Furthermore, there is no integration between the ADAS systems and the road; the only interaction happens through sensors. These limits can be overcome by introducing connectivity between vehicles. For this purpose, an innovative experimental infrastructure is developed within a European project called SAFESTRIP [5]. The project provides the road with sensors and communication devices, with special care on cost and environmental sustainability, to improve the road's safety.
The project's vision is to make roads an active player in the overall safety of mobility. The road must become "self-explanatory" through personalized in-vehicle messages [54], and "forgiving," providing advanced cooperative functions for all the vehicles and users (vehicles with communication equipment and vehicles without), as well as autonomous vehicles (some use cases re not treated here, oe can refer to [5]), PTW as well as pedestrians. Communication between vehicles and "smart" roads may use the so-called Vehicles to Everything (V2X) standard [46] (European version). This technology provides fast and reliable communication between vehicles and between vehicles and the road infrastructure of SAFESTRIP. On the other hand, the V2X requires specific hardware and, therefore, it will eventually exclude certain road users. In SAFESTRIP we are interested in also covering vehicles that are not equipped with such specific hardware. For this type of vehicles we develop a “copy” of the V2X using the Message Queuing Telemetry Transport (MQTT) [91] protocol on LTE connection. We define these two types of vehicles as equipped and non-equipped, respectively.
In this work, we propose a unifying framework for connectivity-based ADAS deployed in the context of the SAFESTRIP project. The framework provides the necessary intelligence to interpret the intention of the road users and provide (personalized and adaptive) warnings. As stated earlier, the agent we develop is built upon the Co-Driver framework. This thesis addresses Co-Driver development and its specializations. The key idea is to create an agent that can exploit both the in-vehicle information and those provided by the infrastructure to predict the driver's intention. It must consider other road users' behavior and continuously monitor the scenario to warn the driver and road users when necessary. The agent is integrated into a flexible framework and allows both to be used in a decentralized and centralized approach improving safety and user acceptance; this framework is the SAFESTRIP system. The first task of the SAFESTRIP system is to detect the vehicles (equipped or not). Once it happens, the system assigns one artificial agent (the Co-Driver) to the vehicle. The Co-Driver follows the vehicle during driving to accomplish specific driving assistance tasks. Moreover, it takes advantage of the scenario information provided by the system. On equipped vehicles, the agent runs on board. If not, the corresponding agent is instantiated on a remote server or "cloud".

In this context, the Co-Driver interacts with the driver through a warning system running on a specifically designed HMI. All the safety applications of SAFESTIRIP project are built upon the Co-Driver agent. The role of the agent is to receive real-time data from the SafeStrip infrastructure (and other vehicles if applicable) and generate warnings for the drivers about the proper behavior w.r.t the danger around. The Co-Driver is based on a two-layer artificial agent. The upper layer is responsible for interpreting the environment and specifically for understanding which high-level behaviors are needed to avoid the danger (i.e., a pedestrian at the zebra crossing correspond only to one safe high-level behavior: to stop before the crossing, or at least being ready to perform a safe stop maneuver up to the pedestrian leaves). The lower layer is more like a low-level driving agent that translates the goals provided by the upper layer into actual human-like maneuvers (longitudinal motion primitives, see 1.3). By comparing such safe actions with the actual human behavior, the Co-Driver can evaluate which (and if to) warning to issue to the driver. Several levels of warning are available, from informatory to high severity.

The safety application developed have been field-tested, and a variety of use cases are selected for their safety-critical level. The use cases are listed below:

Pedestrian protection and wrong-way driving At first, we have the pedestrian crossing use case, known as "VRU" use case. The system can detect pedestrians using sensors (strips) on the zebra crossing. The second use case is about vehicles entering a highway in the wrong way. The SAFESTRIP sensors on the ramp detect such vehicles and send warnings to incoming vehicles. These two "sources of danger" are grouped in the same use case because the logic is similar (A ramp occupied by a wrong-way vehicle is considered a static obstacle).

Roadworks Roadworks, requiring lane level personalization, is another use case representing a good benchmark for the SAFESTRIP system and the Co-Driver. In short, vehicles on open and closed lanes must receive different warnings.

label The last use case concern the intersections. It is the most critical to manage because it involves pieces of information from other vehicles. However, it is also one of the most interesting. The Co-Driver is here employed with a higher level of complexity (we must consider the right of way and the crashes between vehicles during the crossing).

These use cases share the same agent architecture with some differences that are mainly related to the type of danger and the integration in the equipped roads. Field validation is performed for all the use cases, both for equipped vehicles that are provided with V2X technology and for non-equipped vehicles that uses a smartphone for the HMI and low-cost RFID tags for identification on the equipped road.

The Co-Driver framework that we just defined for this safety application, can accomplish other tasks imitating drivers' behavior. The ability of the Co-Driver framework to predict the driver intention can be used to estimate the speed profile that drivers will follow in the next seconds. As already discussed, the Co-Driver is a bio-inspired agent (See Section 1) that mainly aims at modeling and imitating a human driver so that it can be exploited for future velocity profile prediction. Therefore we created a new Co-Driver instance which renders itself as a new, flexible, and computationally lightweight algorithm for velocity prediction. The prediction outcomes are validated with naturalistic driving data and compared with other prediction methods from the literature.

We combined the Co-Driver instance that estimates the driver's intended velocity profile with a Model Predictive based energy management system that optimizes the power split in a Hybrid Electric Vehicle.
We developed a novel Model Predictive Control approach, including a state-of-the-art electrochemical battery model that is used in an online Model Predictive Control (MPC) for the first time. The validation is performed with driving data while the power train and the vehicle dynamics are simulated using actual vehicle data and efficiency maps. Thanks to the accurate battery model, the cost function of the OCP that is used in the approach contains a term related to the battery lifespan (capacity loss), allow to have an optimal trade-off between the battery degradation and the fuel consumption. The battery degradation is a significant aspect since severe battery usage may result in an early replacement. Mitigating the ecological and the economic benefits introduced by the Hybrid power-train.

The last topic that we address in this thesis concerns mobility efficiency. We bring back the concept of connected vehicles that we exploited in the safety applications of the Co-Driver to extend it to autonomous vehicles. This application was inspired by the management of the intersection of the third SAFESTRIP use case combined with a novel approach used to optimize airport traffic [13]. The problem is formulated as a bi-level optimization problem. In the lower level, the motion of the vehicles is calculated as the solution of an optimal control problem that maximizes the final velocity when approaching the intersection in a given time window. Instead, the higher level provides the time to arrive at the intersection to all the vehicles involved. This higher-level problem eventually includes the optimization of the vehicle order. The latter case involves integer variables that cast the problem into the family of mixed integer linear problems (MINLP). The formulation considers the vehicle's velocity as a function of the time they take to arrive at the intersection. Limits on the vehicle acceleration and limits on the velocity due to curvature of the road are taken into account. To solve this problem, we introduce a powerful linearization technique based on a piecewise linear approximation of the non-linear functions. This technique allows to bound the error on the approximation and even to simplify the problem itself since it encloses the case-defined function into its formulation. The framework is validated in simulation in a complex intersection. The problem can be used in any geometry of the intersection as soon as the overlapping portion of the lanes is known.

Structure of the thesis

This thesis is divided into 4 chapters.
The first introduces the Co-Driver concepts and the motion primitives, which will be the building blocks of the application explained in **Chapter 2** and **3**. **Chapter 2** introduces the SAFE STRIP project and describes the safety applications developed and tested herein. The general application framework is described in detail. In the **Chapter 3** an instance of the Co-Driver to predict drivers' intended velocity is explained and validated and then combined with an online MPC approach we proposed for hybrid electric vehicles. In **Chapter 4** we illustrates the Intersection Management strategy for Connected Autonomous Vehicles. Every chapter presents one or more sections to discuss the existing solutions. In the first chapter, the state-if-the-art is discussed for each use case, since in literature, they are addressed as separate problems.

Chapter 1

The Co-Driver agent

"Every PhD thesis should start with a proper quote"

— *This thesis*

Predicting in advance the intentions of a driver is a critical task in the development of Advanced Driving Assistance Systems (ADAS) up to the application level of automation L3 (according to SAE autonomous driving nomenclature [70]). Correctly inferring the maneuver's goal beforehand allows timely delivery of warning messages and intervention to support the driver to safely and comfortably accomplish his/her desired maneuver. Additionally, a highly accurate estimation minimizes false alarms or, even more critically, missed alarms or interventions, improving the driving experience. The level of assistance has to be tailored to the driver's behavior and way of executing the maneuver.

In this thesis, the applications of driving prediction span from the safety ADAS systems to the optimization of the power-train usage. It is possible because our prediction generates future longitudinal maneuvers, and we can use them for different purposes (generative approach). The safety ADAS systems Are mainly related to the intentions rather than the velocity profile. Understanding drivers' intentions allows to assist the driver and improve his/her maneuver whenever necessary (*e.g.* brake more if driver's safety is at risk). We can define this type of capability as **interpretation** of intentions, which in short is the capability to understand the final goal of a maneuver in advance. Another task that we want to accomplish is to predict the velocity profile in a defined time horizon. Such capability allows, for instance, to quantitatively optimize drive-line response or energy management of hybrid or purely electric vehicles. The generative approach adopted by the Co-Driver can achieve both of these goals. We can define the ability to understand the "high-level" intentions as "interpretation" and the ability to "copy" the maneuver to predict the velocity profile, as **imitation**. We call "mirroring" the concurrent process of imitating and interpreting an action, in our case, a driving maneuver.

1.1 Mirroring, Co-Driver theoretical basis

The ability to interpret and to imitate may be seen as two separate abilities. However, This is not necessarily true. In neuroscience, it was observed that "a supramodal representation of action closely links perception and production of human action." In [86] Meltzoff defines the so-called "like me" framework. This framework defines the simultaneous interpretation and imitation of another agent's behavior introducing the following concept: if someone performs the same action as you, he/she must have similar "internal states" as you. This concept resembles the observers in control theory, where a parallel system imitates inputs and outputs to observe the internal states of another system. Another concept of cognition based on a direct representation of actions is the *Simulation theory of cognition* by Hesslow [64]. In this theory, thinking is conceived as a simulation of action and perception cycles that are carried out as inhibited sensory-motor loops [63]. Motor cognition is then conceived as the execution of covert sensory-motor loop activities. A particularly evident example to understand the concept of simulating sensory-motor loops occurs when one is about to lift a heavy object. If the object is not heavy as expected, one reacts by lifting it too much and getting surprised. In the simulation theory of cognition such astonishment is due to the simulation one made in mind, which does not resemble the real sensory-motor loop.
This approach to cognition was also studied by Jannerod et al. in [74] and [73]. In this cognition framework, the interpretation (and imitation) of others' intentions are eventually a simulation process. An agent that wants to interpret (and imitate) another agent's physical action starts from observing its motion. After that, it tries to reproduce the same motion simulating the sensory-motor loops internally. This process eventually involves both interpretation and imitation. We call it a "mirroring" process [69] [37]. The Co-Driver is based on the concept of mirroring applied to driving. In the next sections, we will start with a general concept of a mirroring agent to explain the Co-Driver. To model a mirroring agent that can generate sensory-motor loops (we will detail such loop for driving in Section 1.3) we use a framework model proposed by Hurley in [69] called Shared Circuit Model (SCM). This model decomposes the sensory-motor loops into two components: the forward models and the inverse models. The shared circuit model resembles what in engineering is called Model predictive control, or predictive control in general. In fact, in [69], the **forward models** are internal models that imitate the outputs expected from the environment. Such a model consists of an expected output mapping on the inputs, and in an organism, such structure can be called "efference copy." In parallel to the MPC approaches in engineering, the forward model can be seen as the dynamical model of the system. This analogy with the Model Predictive Control is more evident in the previous works of Grush [55], and Miall [87]. The forward model represents how the human imagines the feedback (and the effects in general) of the environment caused by the action. Likewise, in the driving framework, the forward model is the ability of the driver to figure out the vehicle response on the road. As the name suggests, the forward models map actions to the feedback from the environment, allowing predicting motion strategies. The predictive strategy improves the response of the agent, and it is essential in human motor control [69]. If the feedback does not match the expectation, one can also refine the forward model or correct actions during executions (remember the lifting-a-weight example). Such adaptive behavior is possible by comparing simulated feedback with the actual one. The step from mere executions to **mirroring** needs another “layer” [69]. The forward model links simulated motor signals to simulated feedbacks, but this process can also be reversed, using the feedback as an input signal that evokes motor signals. We call this link "inverse emulator." Combining these two factors allows an agent to generate complete sensory-motor loops to accomplish a task or imitate another agent. A graphical representation of the shared circuit model is shown in Figure 1.1.

For the purpose of the inference of intention, other remarkable contributions are provided by Demiris et al [38][40][39][28] and Wolpert et al [126][62][128][127][14]. We borrow one methodology from their work to add a missing ingredient to have a complete idea of a mirroring agent. They use the cognitive frameworks based on simulated sensory-motor loops to formalize the mirroring process as a generation of self behaviors and comparison of others' behaviors. In other words, they generate a pool of agent behaviors that are compared with the observed behaviors. This approach is called **generative**.
If we talked about sensory-motor loops in general, we could now link these sensory-motor loops to actual goals that these sensory-motor loops have. The generation of sensory-motor loops is strongly related to the context since they are hypotheses of feasible and optimal actions. To find how to define these sensory-motor loops the concept of **affordance** help us [53]. The concept of affordance is exactly what completes the framework on cognition conceived as actions and simulation of actions. In the theory of affordances [53], they are defined as specific interactions (usually involving motion) between an agent and an object. An affordance is not a property of the agent nor a property of the objects; it is defined in a specific possible interaction between the user and the object. For example, opening a door is an affordance between the human and the door. Such affordance does not

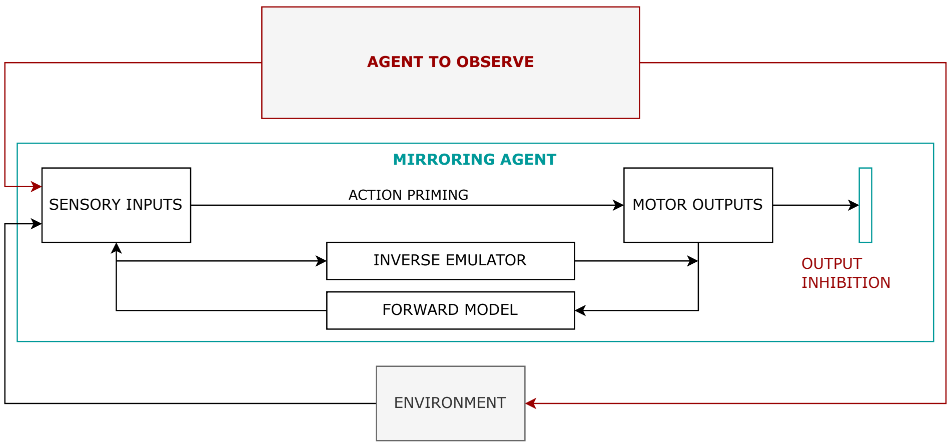


Figure 1.1: Shared circuit model for a mirroring agent. The mirroring agent tries to interpret and imitate the action of the agent to observe by simulating the same actions basing on the feedback from the environment. In short, inverse emulator models the agent behavior while forward model the physics in which the actions occurs.

exist if the door does not have a handle since it cannot "suggest" this affordance. In driving, an affordance may be to follow another vehicle or stop before a give-way sign. Using the affordance concept, the possible actions (sensory-motor loops) are suggested by the context and surrounding objects; we will see how this concept is used in the Co-Driver in driving tasks. This restricts the possible sensory-motor loops that one may generate to imitate another agent since the goal suggested by surroundings are not infinite, making the mirroring process possible as soon as the agents surroundings are shared. Affordances can be instantiated as simulated sensory-motor loops; this is important because more than one affordance at a time is instantiated in cognition. In [30] human action selection is conceived as a competition between potential affordances. A pool of affordances is simultaneously considered, and then only one is executed. Another essential theoretical foundation of the cognitive structure we are going to build on the Co-Driver is the subsumption architecture introduced by Brooks [26]. This theory has a layer of simple sensory-motor loops (or affordances) that compose increasing complexity behaviors. Therefore, the final idea of a mirroring agent is an agent that behaves similarly to the other agents, generating a pool of affordances, consisting of potential (and simulated) sensory-motor loops that mimic the exact interactions between the same interactions agents and the surrounding. Moreover, it must select one that best matches one of the mirrored agents. For implementations such as the Co-Driver, sometimes we do not need to mimic an affordance; we will see that for some applications, we want only to check if a potential affordance of the driver is within a specific safe set. We use this concept to build our Co-Driver agent in the next section.

1.2 The Co-Driver as a mirroring agent for driving

The concept of Co-Driver was born with the main scope to predict drivers' intentions in both of these senses. The theoretical background for the Co-Driver was mainly collected in [32] to design it as a bio-inspired architecture for driving intention prediction used to design an advanced system to support the driver in various driving tasks (i.e., lane keeping, intersection support). An artificial Co-Driver in this context is defined as an agent able both to *drive* like a human and to *interpret* his/her driving intentions [32][35][34][33]. The Co-Driver's ability to drive does not have to be complete. It can be restricted to the pool of affordances we are interested in a specific application. We have already seen a mirroring agent in general, but to understand how we specialize this concept to a Co-Driver is helpful to introduce one example of a "natural" Co-Driver [32]. For instance, we can think about a driving license tutor: the tutor is not only able to *drive* like humans, but s/he also has to anticipate the trainee's actions by *interpreting* their intentions in order to intervene in emergencies to correct a mistake through the auxiliary vehicle controls. In this analogy, the relation between the driver (trainee) and who plays the Co-Driver role (the tutor) is made of a mutual and continuous feedback relation. It is worth noticing that, since the Co-Driver agent is aware of the driver's goals, it actively intervenes only when necessary. Note that the central role of the tutor is to *interpret* what the trainee is doing. However, he/she has also to intervene if necessary, and to do this, affordances on safe behavior are always latent in his pool of possible actions.

Action priming

We decided to design a mirroring agent that uses a generative approach. The core of the mirroring process is then based on the generation of a set of human-like actions. In this work, which is about driving, the affordances are longitudinal maneuvers. The Co-Driver always generates a pool of longitudinal maneuvers of a specific type. In this section, we define these maneuvers, called longitudinal motion primitives.

Affordances

To synthetize driving affordances, we must before understand how humans plan their motion. From 70s, many researcher focused on the criterion that drives the human motion planning: Wolpert et al. [14][93], Harris [60],[61], Viviani and Flash [121], Flash et al. [48]show evidence that biological motion planning occurs according to optimal criteria. In particular, Harris and Wolpert indicate minimization of postmovement errors and rejection of neural noise (the "minimum variance" principle). This underlying efficiency criterion can explain the observed human time–accuracy tradeoff and the so-called "two-thirds power law" (see below). Optimal Control is thus a convenient strategy that reproduces simple sensory-motor primitives. Moreover, the application of OC within a receding-horizon scheme explains another observed fact, known as the "minimum intervention principle" [118], [113], which states that task-irrelevant deviations are left uncorrected. Viviani and Flash [48], [49], [24][23] and others also showed that lateral acceleration and speed in human movements are inversely correlated with curvature (the "two-thirds power law"), which Harris explains as a consequence of the minimum variance principle. Furthermore, ordinary drivers (not professionals for racing) use lateral acceleration and speed in inverse correlation with road curvature far from the physical limits. This phenomenon originates from the minimum variance principle. Drivers reduce speed in curves to keep sufficient accuracy in lateral position [23]. In [24] the authors use this relation to develop an agent that can provide warnings using the mirroring process on curves for motorcycles (similarly to our Co-Driver). In the next section, we introduce longitudinal motion primitives considering the affordances in the Co-Driver context. Those are the building blocks for the synthesis of driving behavior for the mirroring process and the Co-Driver.

1.3 Longitudinal Motion Primitves

The longitudinal motion primitives are the essential component of the Co-Driver. They represent atomic maneuvers that a driver can plan and pursue. The objective of these maneuvers is to provide a human-like link from the current vehicle state to a specific goal of driving (e.g., reaching the desired velocity), thus being the **affordances** for the Co-Driver. The longitudinal motion primitives directly derive from the minimum jerk principle applied to driving (See Section 1.2). Here, the longitudinal jerk is the control variable. We also assume that human drivers plan the longitudinal motion of the vehicle following "simplified" dynamics [18]. In short, we assume that a driver can control the vehicle tracking the desired jerk.

1.3.1 Primitives notation

We will use longitudinal motion primitives frequently. Therefore, we need a precise notation. Therefore, let us now formalize the Generation of a primitive (or a set of them)

Primitive generation

We need to distinguish between a function that generates a motion primitive and an instance of a primitive itself (or a set of primitives). The generation function is *always* indicated by $\mathcal{M}_{\texttt{type}}(arg_1, arg_2, ...)$, where `<type>` has to be replaced with the acronym of the primitive type (i.e. CSS). While the arguments $arg_1, arg_2, ...$ are defined in order. Arguments differs between different types of primitives. A function $\mathcal{M}_{<\text{TYPE}>}(...)$ is an abstraction that is used to generate a primitive *instance* by means of the arguments of the function.
The arguments are the initial conditions, the final conditions, and some parameters if needed. An instance is an actual primitive that represents an evolution of the states from the initial conditions up to the final ones (in time).

Note that the initial conditions (velocity and acceleration) are always required to generate a primitive. Let us now take a general expression of a primitive instance generation.

$$m_{<\text{instance name/index}>} = \mathcal{M}_{\text{FFCT}}\left(v_0, a_0, v_f, T\right). \tag{1.1}$$

The FFCT type of primitive is explained in Section 1.3.4. It requires the initial acceleration and velocity, final time T and the final velocity v_f as parameters in this order. In the example (1.1), $m_{example}$ represents only *one* FFCT primitive with v_0 and a_0 as initial conditions, v_f as the required final condition, and T as a required parameter. The argument T is a required parameter and it is the duration of the primitive. When we generate a single primitive we are going to use lower letters (like *m* in (1.1)), for set of primitives we use capital letters.

Generation of sets of primitives

In some applications, we are interested in generating sets of primitives. To generate a set, it is enough to call a primitive generation with one or more parameters defined as a set instead of a single value. To provide an immediate example, we generate a set of the same type of primitive seen in Equation (1.1). Then, in Equation (1.2) we generate a set of all the primitives of type FFCT that have the duration between T_0 and T_1. This set of primitive is continuous. Therefore, we can evaluate only a single instance of it (or more than one), assigning a value to T in the domain $[T_0\ T_1]$.

$$M_{<\text{instance name/index}>} = \mathcal{M}_{\text{FFCT}}\left(v_0, [T_0\ T_1]\right). \tag{1.2}$$

In Chapter 2 we use sets of primitives, and we need to evaluate only the two primitives (or even only one) that are at the boundary of the set.

1.3.2 Vehicle-driver simplified model

The primitives' expression is always calculated as a closed-form solution of an optimal control problem. The first thing to define is the model used, which is the same for all the types of primitives.
The idea behind this dynamical model is that a driver can impose the longitudinal motion of the vehicle by the pedals (this is essentially the forward model seen in Section 1.1). The driver does not use a mathematical model of the vehicle, but s/he can control the vehicle. The scope of the primitives is to provide the future longitudinal motion planning of the driver. Therefore we are interested in modeling the closed-loop vehicle-driver, controlled by the driver. Using the curvilinear coordinate $s(t)$ as a state, the following equation describes the model:

$$\left\{\begin{array}{rcl} \dot{s}(t) & = & v(t) \\ \dot{v}(t) & = & a(t) \\ \dot{a}(t) & = & j(t) \end{array}\right. . \tag{1.3}$$

The model is a triple integrator since the system's input is the longitudinal jerk $j(t)$. The states $a(t)$ and $v(t)$ are the longitudinal acceleration and velocity, respectively, while $s(t)$ is the curvilinear coordinate along the vehicle's trajectory. In Section 1.2 we discuss the choice of the jerk as the input variable. The primitives are characterized by their final condition and if the final time is constrained or not. In this work, we define only primitive where the duration is constrained because they were suitable for this application. In [116] the authors also use primitives with unconstrained time for velocity prediction.

Primitives representation

The primitives represent the evolutions of 3 states and 1 control indicated by $s(t), v(t), a(t), j(t)$ respectively, starting from the current state of the vehicle. All the longitudinal motion primitives are subject to the model (1.3) and their initial conditions are known. It is enough to know the evolution of only one state (or control) to retrieve the others. By convention, unless otherwise specified, we represent a primitive by the curvilinear position function $s(t)$. The coordinate $s(t)$ is the curvilinear coordinate along the vehicle's trajectory. One reference for curvilinear coordinates can be found in [83].
As stated earlier, we assume a human driver controls the vehicle by its jerk $j(t)$. Intuitively, one can imagine the jerk as proportional to the rate of the gas pedal (or the brake pedal) position. The control defined at the very beginning of the maneuver plays a unique role. It is called **initial jerk** j_0. This quantity is the instantaneous control required to start a maneuver. It is used as a metric for the effort required from the driver to start the maneuver [32], in the text, sometimes is referred to it as "effort." Since this quantity is significant, we are interested in mapping primitives to its initial jerks. The initial jerk of a maneuver can be defined with the following notation:

$$\begin{aligned} j_0(m_1) &= j_{m_1}(0) \\ J_0(M_1) &= j_{M_1}(0) \end{aligned} \tag{1.4}$$

We are also interested in evaluating a set of initial jerks coming from a set of primitive. In such a case, the capital letter is used. In Equation (1.4), the notation is depicted. While evaluating the initial jerk of a single primitive is straightforward, evaluating a set of initial jerks is less immediate. We are interested in defining such sets only for specific sets of primitives. The sets are detailed in their specific sections.

Primitives evaluation

The final scope of the primitives is to represent some motion of a vehicle. By their closed form expression, we can retrieve the values of the states. Few aspects must be pointed out.

- The domain of a primitive is always $D = [0, T]$ therefore: $t \in [0, T]$.
- The initial curvilinear position is *always* 0. Therefore $s(0) = 0$.
- The states and controls are *always* indicated by the same symbols s, v, a, j. Pedixes and other identificators are allowed: (i.e $\tilde{v}$,j_i)

The notation of a primitive's evaluation is composed of the state's (or control's) name and a pedix indicating which maneuver it belongs to. For instance, if we generate a maneuver m_1, and we want to evaluate the velocity at time $t = t^*$, we can write $v_{m_1}(t^*)$ or simply $v_1(t^*)$ (if it does not generate ambiguities in its context).

1.3.3 Longitudinal Motion Primitives types

The primitives are the units of the Co-Driver longitudinal driving. We assume that the drivers plan the control input for the entire duration of one motion primitive. However, the driver can plan for several primitives simultaneously to open for several choices (affordance competition). The primitive is formulated as an optimal control problem that makes use of the model in (1.3) and of a cost function that reflects the minimum jerk principle (and the minimum variance principle for this specific model). The vehicle's state gives the initial conditions of the primitives, and the final conditions depend on the goal of the primitive. Such primitives can represent atomic motion planning for driving intention, and the final goal is the thing that most characterize a primitive type.

1.3.4 Free Flow Constrained Time (FFCT)

The first type of primitive we illustrate is called Free Flow Constrained Time (FFCT). This primitive aims to model the driver's intention to reach a velocity v_f in a predefined amount of time T. This primitives wants to model just the simple action to reach a desired velocity, the duration of the maneuver T in this case, serves as parameter for

the "urgency" of the primitive. The shorter the time, the more urgent the primitive and the higher the effort.
The velocity is indeed the target of the primitive, as well as the duration. The driver's intention here is to reach the desired velocity in a time T, in the literature other approaches is to use a weight on time in the cost function [116][32]. We decide to use the duration as a parameter. The main advantages of constraining the time are two. Firstly, The duration T is easier to tune since it has a plain meaning. Secondly, the solution is in a completely closed form (see (1.7)), and do not require solving polynomials. On the other hand, this primitive has the disadvantage of not representing a realistic maneuver if we choose an unreasonable T. For instance, selecting a short time T for high-velocity differences ($v_f - v_0$) lead to high accelerations and high jerks, even unrealistic values.
Let us now show the boundary condition of the optimal control problem:

$$\begin{cases} s(0) &= 0 \\ v(0) &= v(t_0) \\ a(0) &= a(t_0) \\ v(T) &= v_\mathrm{f} \\ a(T) &= 0 \end{cases} . \tag{1.5}$$

As we have seen for the other primitives, the initial position $s(0)$ is always zero since the primitive always represents a longitudinal motion starting from the current curvilinear position. The final velocity v_f and the duration T are the degrees of freedom of this primitive. We can now define the optimal control problem.

$$\begin{gathered} \min_{\mathrm{j(t)}} \int_0^T j^2(t) \\ \text{subject to (1.3), (1.5)} \end{gathered} . \tag{1.6}$$

The solution of (1.6) is hereby reported in term of $s(t)$. The states and the controls can be easily retrieved through its derivatives using Equation (1.3). The expression of $s(t)$ is a fourth-order polynomial:

$$p(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4. \tag{1.7}$$

The coefficients of the polynomial (1.7) depend on T, that is, the duration of the maneuver. The coefficients are:

$$\begin{aligned} c_0 &= 0 \\ c_1 &= v_0 \\ c_2 &= a_0 \\ c_3 &= \tfrac{v_\mathrm{f}-v_0}{T^2} - \tfrac{2a_0}{3T} \\ c_4 &= \tfrac{v_0-v_\mathrm{f}}{2T^3} + \tfrac{a_0}{4T^2} \end{aligned} . \tag{1.8}$$

In Figure 1.2 some instances of primitives are illustrated. We have two plots. Both of them show the velocity, the first versus time, and the second versus the curvilinear coordinates $s(t)$. The primitives in the example belong to the following set:

$$M_{FFCT,examples} = \mathcal{M}_{FFCT}(7, 1, \{2, 5, 8, ..., 29\}, \{4, 6, 8, 10\}). \tag{1.9}$$

We hereby recall the importance of the **initial jerk** $j_0 = j(0)$. It is proportional to the instantaneous effort required to pursue a primitive. The initial jerk of a FFCT can be computed from the third derivative of the polynomial in Equation (1.7). The third derivative is $j(t) = 6c_3 + 12c_4 t$ and replacing $t = 0$ we get $j_0 = 6c_3$ that results in:

$$j_0 = \frac{v_\mathrm{f} - v_0}{T^2} - \frac{2a_0}{3T}. \tag{1.10}$$

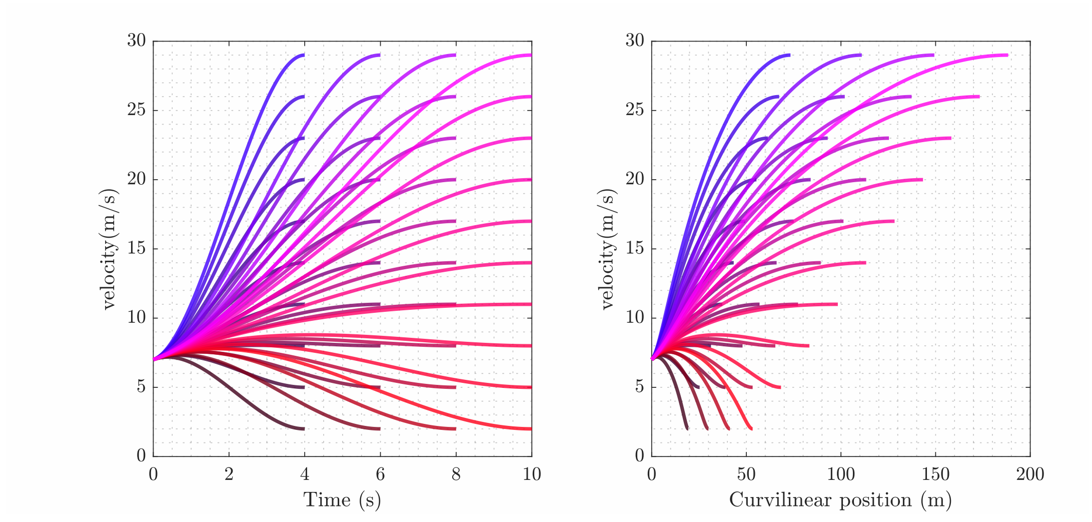


Figure 1.2: Example of Free Flow Constrained Time primitives. On the left velocity versus time, on the right the same primitives represented on a velocity-position. The ste of primitive in this example is provided in Equation 1.9

.

In some applications it is useful to retrieve the final velocity starting from an initial jerk. To get v_f from j_0 expression (1.10) can be inverted to get:

$$v_f = v_0 + \frac{4aT + j_0 T^2}{6}. \tag{1.11}$$

The application of this primitive is mainly related to the prediction scheme of Chapter 3. Therefore, we are not interested in finding any set of initial jerks, since the application does not require one.

1.3.5 Constrained Space Stop (CSS)

This is the maneuver used to model the driver intention to stop. This intention can be modeled with different levels of accuracy [35]. However, we aim at modelling the “main phase” of the maneuver to perform the mirroring of intention. In our application of this maneuver (See Chapter 2) we are interested in the effort necessary to start a braking maneuer, not to model in detail the velocity profile in the braking phase. The initial conditions of the maneuver are velocity and acceleration as usual, while this time the final distance s_f is the final condition, and the final velocity is always zero. being a maneuver to stop, also the final acceleration is zero. The time T is the duration of the maneuver and it is not free, the expression is in (1.16) . Initial and final condition reads as:

$$\begin{cases} s(0) & = & 0 \\ v(0) & = & v(t_0) \\ a(0) & = & a(t_0) \\ s(0) & = & 0 \\ s(T) & = & s_{f,0} \\ v(T) & = & 0 \\ a(T) & = & 0 \end{cases}. \tag{1.12}$$

The Optimal control problem is analogous to the FFCT, except for the boundary conditions.

$$\begin{gathered}\min_{\mathrm{j(t)}} \int_0^T j^2(t) \quad . \\ \text{subject to (1.3), (1.12)}\end{gathered} \tag{1.13}$$

The solution of (1.13) is hereby reported in term of $s(t)$. From tis derivatives, we can easily retrieve the other states and the control. The expression of $s(t)$ is a fifth-order polynomial:

$$p(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 - c_5 t^5. \tag{1.14}$$

The coefficients of the polynomial (1.14) depend again on T, that is, the duration of the primitive. The coefficients read as:

$$\begin{array}{lcl} c_0 & = & 0 \\ c_1 & = & v_0 \\ c_2 & = & a_0 \\ c_3 & = & 10\frac{s_f}{T^3} - 6\frac{v_0}{T^2} - \frac{3a_0}{2T} \\ c_4 & = & -15\frac{s_f}{T^4} + 8\frac{v_0}{T^3} + \frac{3a_0}{2T^2} \\ c_5 & = & 6\frac{s_f}{T^5} - 3\frac{v_0}{T^4} - \frac{a_0}{2T^3} \end{array} \quad . \tag{1.15}$$

For this primitive, the time T is a not an argument, but it depends on the final distance as well, it can be retrieved with this expression:

$$T = \frac{10 s_f}{2v_0 + \sqrt{4v_0^2 + 5a_0 s_f}} \tag{1.16}$$

In Figure 1.3 some instances of primitives are illustrated. For this type we have no parameter but the distance. In Figure 1.3 we show several examples of CSS primitives, varying initial and final conditions:

$$M_{CSS,example} = \mathcal{M}(\{6, 8, 10, 12\}, \{1\}, \{5, 7, .., 20\}). \tag{1.17}$$

This type of maneuver is threated also in [34], to apply a similar concept to intersections, and in [85] for smart

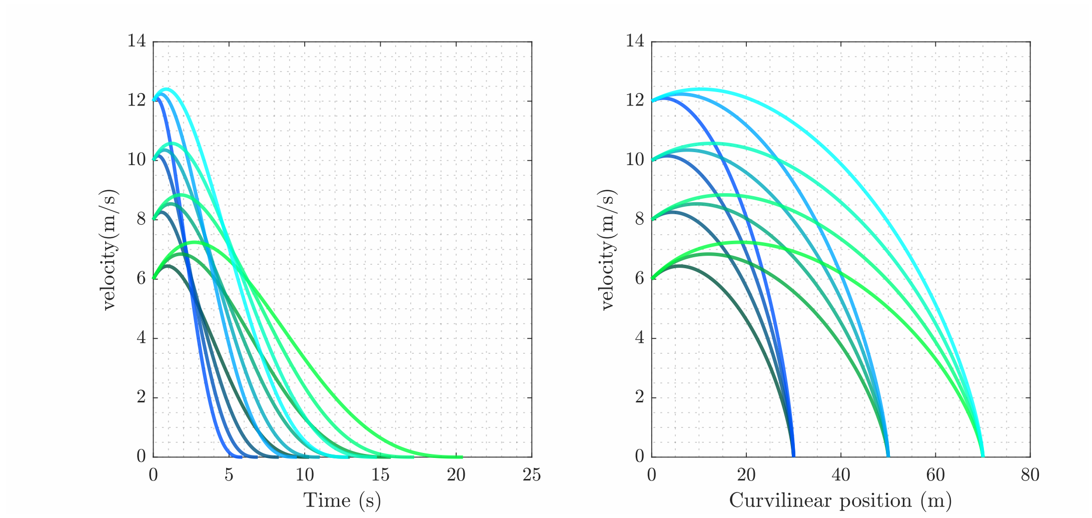


Figure 1.3: Example of Constrained Space Stop primitives. On the left velocity versus time, on the right the same primitives represented on a velocity-position plan. Initial velocity and final position are varied, as shown in Equation (1.17)

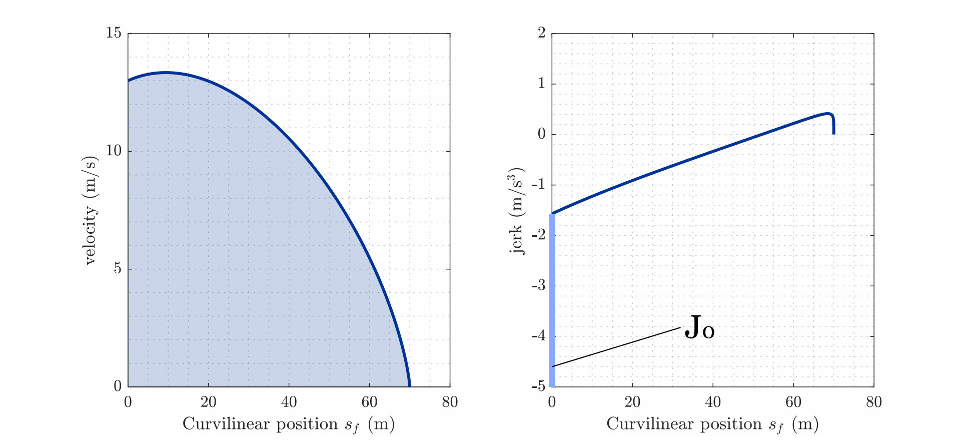


Figure 1.4: Example of initial jerks sets generate from a M_{stop} primitives set. The primitives' set comprehends all the maneuvers that stops before $s_{f,0}$. To compute the jerk set, we need only the jerk of the CSS maneuver with $s_f = s_{f,0}$ since the set is $J_0 = (\, -\infty \, , \, j_0(\mathcal{M}_{css}(., s_f = s_{f,0}))\,]$. The jerk set J_0 is indicated in light blue.

traffic lights.

jerk set of Constrained Space Stop primitives

The jerk set is relevant for this type of primitives. We need to discuss how can we compute the jerk set given a specific set of CSS primitives.
The goal to **stop before a certain distance** s_f, is modeled with the following set of CSS primitives M_{stop}:

$$M_{stop} = \mathcal{M}_{CSS}\left(v_0, a_0, (0 \;,\; s_f]\right). \tag{1.18}$$

The set M_stop represent the possibility to stop at s_f or *before*, up to the limit scenario in which the distance to stop is 0. This set is used to retrieve all the possible safe maneuvers when one have to stop before a certain distance.
For this case, in Chapter 2 we are interested in finding the set of jerks associated to the set M_{stop}. To retrieve the set, we must before define a property of the initial jerk of the CSS pimitives:

$$\forall s_{f,1} < s_f f, 2 \;,\; j_0(\mathcal{M}_{CSS}(v_0, a_0, s_{f,1}) < j_0(\mathcal{M}_{CSS}(v_0, a_0, s_{f,2})). \tag{1.19}$$

The initial jerk j_0 in monotone with the distance s_f of the primitive. This means that now we can calculate the set of initial jerks associated with M_{stop}.
The set pf the initial jerks is upperbounded by the initial jerk of the "slower" primitive (the one to stop *exactly* at s_f), and it is not lower bounded, since we want to consider all the possible primitives that stops before s_f.

$$J_0((\mathcal{M}_{CSS}\left(v_0, a_0, (0 \;,\; s_f]\right)) = (-\infty \quad , \quad j_0(\mathcal{M}_{CSS}(v_0, a_0, s_f)] \tag{1.20}$$

In Figure 1.4 we show an example of a initial jerk set for the above case.

1.3.6 Constrained Space Time (CST)

This primitive links the initial conditions to a final position at a certain time T (that is a parameter). It is used to model the intention to cross a point (at a s_f distance) in a certain amount of time T.
The final velocity is a function of the final time. The relation between T and final velocity can be inverted if

needed [34].

$$\begin{cases} s(0) &= 0 \\ v(0) &= v(t_0) \\ a(0) &= a(t_0) \\ s(0) &= 0 \\ s(T) &= s_{f,0} \\ v(T) &= 0 \\ a(T) &= 0 \end{cases} . \tag{1.21}$$

The final velocity as a function of time reads as:

$$v_f = \frac{15}{8T} - \frac{a_0 T}{8} - \frac{7a_0}{8} \tag{1.22}$$

The Optimal control problem is analogous to the CSS and FFCT, except for the boundary conditions.

$$\begin{array}{c} \min_{\mathtt{j(t)}} \int_0^T j^2(t) \\ \text{subject to (1.3), (1.12)} \end{array} . \tag{1.23}$$

Once again, the solution of (1.23) is hereby reported in term of $s(t)$. The states and the controls can be easily retrieved through its derivatives using Equation (1.3). The expression of $s(t)$ is a fifth-order polynomial:

$$p(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 - c_5 t^5. \tag{1.24}$$

The coefficients of the polynomial (1.14) depend on the duration of the maneuver T. The coefficients are:

$$\begin{array}{lcl} c_0 &=& 0 \\ c_1 &=& v_0 \\ c_2 &=& a_0 \\ c_3 &=& 10\frac{s_f}{T^3} - 6\frac{v_0}{T^2} - \frac{3a_0}{2T} - 4\frac{v_f}{T^2} \\ c_4 &=& -15\frac{s_f}{T^4} + 8\frac{v_0}{T^3} + \frac{3a_0}{2T^2} + 7\frac{v_f}{T^3} \\ c_5 &=& 6\frac{s_f}{T^5} - 3\frac{v_0}{T^4} - \frac{a_0}{2T^3} - 3\frac{v_f}{T^4} \end{array} . \tag{1.25}$$

In Figure 1.5 some instances of primitives are illustrated. they are the following set, obtained varying the time and the final position:

$$M_{CST,example} = \mathcal{M}(7, 1, 4, 6, 8, 10, 40, 60, ..., 100) \tag{1.26}$$

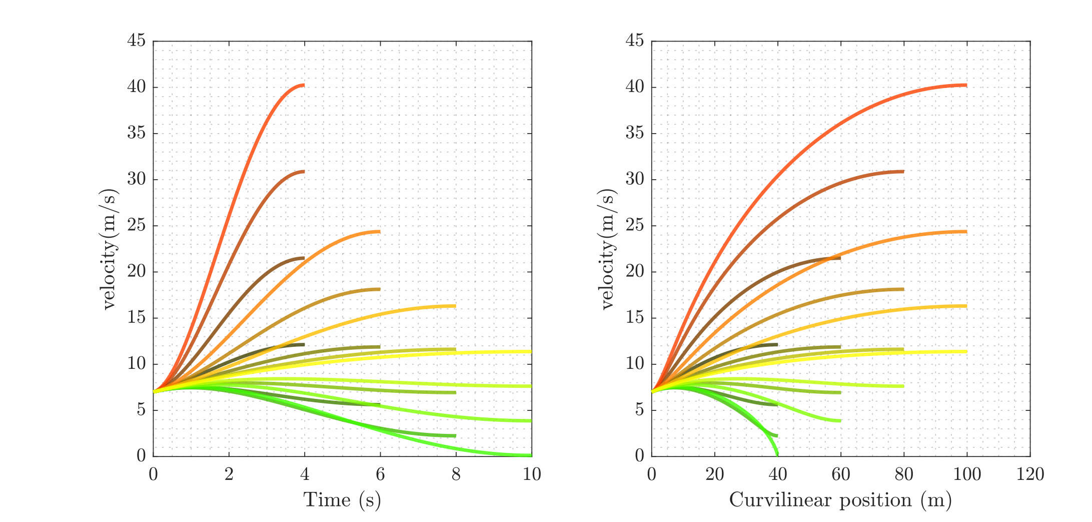


Figure 1.5: Example of Constrained Space Time primitives. On the left velocity versus time, on the right the same primitives represented on a velocity-position plan. Final time and distances are varied. The primitives in the plots are generated in Equation (1.26).

We are again interested in evaluating the jerk sets, to be used in 2. In this case we want to evaluate jerk sets for the following sets of primitives:

$$M_{pass} = \mathcal{M}_{CST}(v_0, a_0, s_f, [T_1 \ , \ T_2]). \tag{1.27}$$

The set M_{pass} is used to represent a passage in a specific time slot given by the compact set $[T_1, T_2]$. Similarly to what we have seen for CSS in Equation 1.19, we have a monotonic relation between j_0 and T:

$$\forall s_{f,1} < s_{f,2} \ , \ j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T_1) < j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T_2)). \tag{1.28}$$

This relation allows us to easily compute the jerk set by using the jerks on the boundary corresponding to T_1 and T_2.

$$J_0 = [\ j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T_1)) \ , \ j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T_2))\ ]\,. \tag{1.29}$$

In Figure 1.6 we provide an example of a set with two CST primitives that deliminate a set between two values of the parameter T, the corresponding jerk is indicated as J_0. The example is generated with the following values: $v_0 = 6$,$a_0 = 1$,$T = [\ 7\ ,\ 14\ ]$. This type of sets is used in Chapter 2. The longitudinal motion primitives will be used for the Co-Driver applications. Both sets and single primitives. Although other types of maneuvers can be defined, we limited this chapter to those used in the thesis.

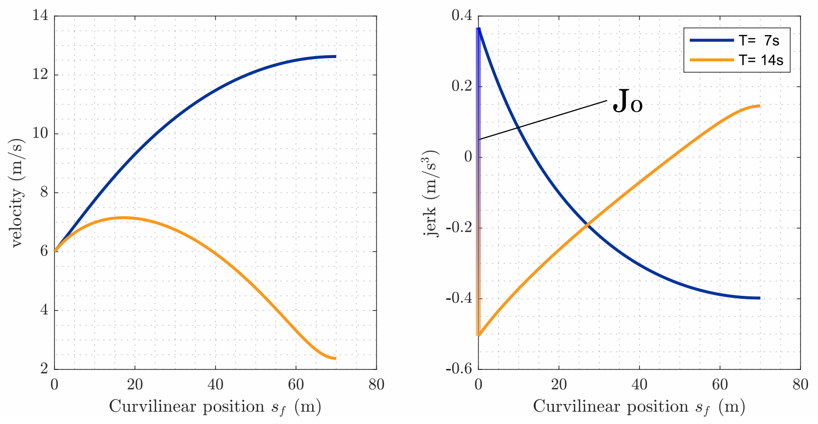


Figure 1.6: Example of initial jerks sets generate from a M_{pass} primitives set. The primitives' set comprehends all the maneuvers between the two maneuvers at the bounds. To compute the jerk set, we need only the two initial jerks of the CST primitives with $T = 7$ and $T = 14$ since the set is $J_0 = [\, j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T = 7))\ ,\ j_0(\mathcal{M}_{CST}(v_0, a_0, s_f, T = 14))\,]$.. The jerk set J_0 is indicated with the shaded vertical segment.

Chapter 2

ADAS systems based on Co-Driver and SAFESTRIP

This chapter addresses the safety ADAS systems based on the Co-Driver paradigm and the SAFESTRIP road infrastructure. The chapter is structured as follows: In Section 2.1 we detail the SAFESTRIP project for the topics that concern this thesis, with a focus on the infrastructure used by the developed ADAS systems. For detailed coverage of the project, the reader can refer to the public deliverables [5] and for a more condensed treatment to [54]. In Section 2.2 the general structure of the Co-Driver for the safety ADAS systems is detailed. In Sections 2.3, 2.4 and 2.5 the ADAS systems are detailed, in each of this section there are in this order: an overview of the existing solutions for the specific use case, the description and the Co-Driver implementation for the use case. In Sections 2.4 and 2.5 some use-case specific aspect of the results are discussed. In Section 2.6.1 we report the aggregated results, and we discuss them.

2.1 Intelligent roads: SAFESTRIP project

SAFESTRIP is a H2020 EU-funded project [5]. It proposes a revolutionary C-ITS technological solution to face several vehicle safety challenges. The project proposes a system mainly based on custom on-road platforms, called "strips." On these platforms, sensors are deployed to get information from the road. However, the strips do not communicate directly with the vehicles. Some component in between is required, as discussed later on. Such components are meant to use Infrastructure to Vehicle (I2V) and Vehicle and Infrastructure (V2I) communication technologies (commonly referred to as V2X) Each road lane is provided with these strips covered by a road marking paint to blend in visually. On the side of the lanes, there is equipment for communication with vehicles. The vision behind the concept of these strips is to make the road "self-explanatory" in order for cooperative ADAS systems to access information related to infrastructure and the scenario in general, comprehensive of other road users. The strips transmit static and dynamic real-time information using a communication gateway (Road Side Unit). Road conditions, kinematics of the vehicles passing (velocity and position), and environmental attributes are exchanged with other vehicles and traffic operators. Such set of information can be intended as “enhanced Dynamic Local Map” eDLM [16]. All vehicles are meant to be included to take benefit of the strips, vehicles that are equipped with C-ITS features (onboard HMI and V2X connectivity) and vehicles that are not. The difference between these types of vehicles is exploited in 2.1.2. The paradigm introduced by the project is an opportunity to introduce safety C-ITS functions with reliable, accurate, and lane-specific information. These functions are based on the CoDriver, and they are designed to deal with other safety-critical scenarios explained in the following sections. The Co-Driver-based ADAS had a double role in the project: being innovative cooperative ADAS

systems and validating the potentiality of the SAFESTRIP infrastructure. In this chapter, we cover the first aspect. We look at the SAFESTRIP infrastructure as a platform to validate the proposed CoDriver-based ADAS. While the infrastructural aspects (and innovations) of the project are only briefly discussed in this section to have an overview of what is necessary to understand the ADAS applications.
Furthermore, we will discuss some aspects of the project that affected the design and the development of the Co-Driver based ADAS systems. For instance, the vehicle classification within the project. This section must be considered a description of a subset of the entire project since the project covered many more aspects like environmental monitoring, pavement monitoring, personalized traffic-related messages. One reader that is interested in the project is recommended to check the documents available from the European commission in [5].

2.1.1 The SAFESTRIP infrastructure

The integrated solution consists of the infrastructure-based platform and the sensing, communication, and energy harvesting custom. Although the solution is explained in [54], in this thesis, we are interested in the parts of the system that are related to safety functions. Therefore, the aspects regarding energy harvesting and added value services (virtual toll gates and road maintenance) are not discussed in this text. To discuss the SAFESTRIP infrastructure in detail, we can start from the system's objective. That are briefly listed below:

- Embed, update, and transmit static information on the road: i.e., the geometry of the road, map data, speed limit, curvature, asphalt characteristic.
- Receive, process and transmit dynamic information of the road (i.e., traffic info, roadworks, level crossing's state).
- Measure dynamic environmental parameters (i.e., temperature, humidity, ice, gas, ambient light) that can affect the safety of the road itself.
- Sensing all the road actors, especially Vulnerable Road Users (pedestrians), in order to provide real-time data for the ADAS systems.

The system's architecture is composed of 4 different connected parts with a well-defined role.

On Board Unit The vehicle (and the driver) are the main actors for ADAS design and deployment. The Onboard Unit (**OBU**) is the part of the system that is connected with the vehicle and the driver. For the latter, using a Human Machine Interface (**HMI**). this unit differs from equipped and non-equipped vehicles (see Section 2.1.2). For equipped vehicles, the OBU can read values from the CANBus, and the HMI is embedded in the vehicle dashboard. For the non-equipped vehicle, the ORU consists of a smartphone connected through LTE to the system using the MQTT protocol and is mainly demanded only to receive information from the system for the HMI.

On Road Unit and Strips The name "On-Road Unit" indicates all the installed components on the road. In this context, the central On-Road Unit is the "Strip," which can detect the vehicles and measure their velocity (not the acceleration). An additional on-road unit is placed on the side of the road to store part of the electronics and humidity and temperature sensors. In this work, we refer to the ORU as the strips.

Road Side Unit The Road Side Unit plays the role of the system's gateway. All the peripheral equipment refers to this communication gateway. It is custom-made and provided with energy harvesting modules (even if it is still connected to the power grid to be reliable). This module communicates with the Strips using Bluetooth technology. The communication takes place only from strips to RSU since the Strips (the ORU in general) are aimed to provide data from the road and the vehicles that pass. Such a unit is equipped with V2X technology and an LTE connection. We will see that these two communication technologies coexists to cover the so-called "equipped and "not-equipped" vehicles.

At the end of this section, to summarize, in Figure 2.2 the SAFESTRIP infrastructure is depicted, with the type of vehicles and all the components. As stated earlier, we represented only the components directly involved in the safety functions. We also discuss some sensors that are not involved in the safety applications for the strips, but they are worth mentioning to understand the strips' capabilities.

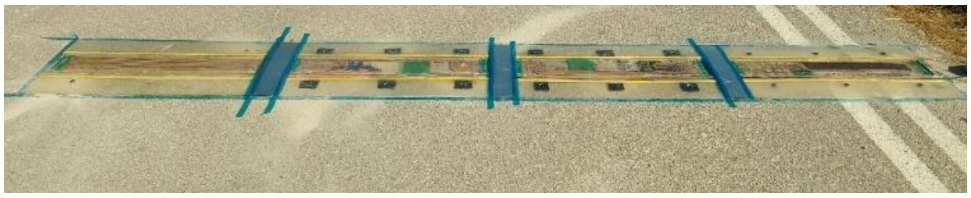


Figure 2.1: Prototype of Strip without the special covering paint

The core novelty of SAFESTRIP lies in the ORU, in particular, the strips.
They are designed to fulfill some critical requirements. They need to have a low energy demand for implementability reasons and sustainability of the solution. They need to be thin enough not to interfere with the road surface. Moreover, they need to be robust mechanically to bear vehicles passing forces. They have to encapsulate the following components:
The sensorial board: it gathers all the data from the various sensing enablers, creates and finally sends in the appropriate format the corresponding message information to the RSU via Bluetooth Low Energy (BLE) wireless communication technology. The implemented version integrates a humidity and temperature sensor and a gas sensor, and a custom Vehicle Detection and Identification System (VDIS) described below, which is the one we are interested in the most. A gas sensor has been developed in the project and is based on the Odorant Binding Protein (OBP). The sensor is intended to infer the presence of gasoline or lubricants on the road in the strips' vicinity. Benzene's key target gas, but other aromatic hydrocarbons (VOCs) are involved. Benzene is found in the emissions from coal, oil, and gasoline. Due to the C6H6 molecule's volatile nature and the high relative weight, this gas stands close to the soil for the ORU benefit. The sensing principle is the electrical impedance change using Odorant-Binding Proteins as molecular recognition elements. The environmental data collection (temperature, humidity, and gas) takes place every 60 seconds while an event-triggered message is produced whenever the system interacts with a passing vehicle enabling the VDIS system. The Vehicle Detection and Identification System (VDIS): it is also a custom system based on mechanical switches manufactured to detect vehicles:

1. position in the lane
2. speed
3. direction
4. longitudinal angle of velocity
5. vehicle type (two-wheeled, four-wheeled) and identification through a Radio–Frequency (RFID) technology implementation.

The energy management module: it is mainly responsible for powering all the individual modules in the ORU by integrating energy harvesters and batteries. Harvesters and batteries (rechargeable and no-rechargeable are supported simultaneously) are connected over an energy management board that incorporates the energy management ICs that hierarchically use the available power sources. In particular, energy is consumed, if available, in the following order: directly from the harvesters (any energy surplus is used to charge the batteries), rechargeable batteries and, no-rechargeable batteries. The energy management module can potentially support different configurations regarding the number and type of the energy harvesters (i.i solar, piezo, radiofrequency), different numbers of batteries per type, external powering, and optional cabling in which two ORUs can share a shared energy storage pool in cases that the operational load is not balanced (i.e., one out of two ORUs is overloaded by a significative amount of more vehicles passing). **Strain gauges:** a strain gauges module is hosted in the ORU to monitor the strain profile of the pavement through two strain gauges per lane, placed at a right angle to each other. Strain gauges accommodate the road predictive maintenance application that has been developed.
The encapsulation layer: It is responsible for the mechanical resistance. The encapsulation layer on top of the ORU aims to protect the ORU electronic components, batteries, sensors, and antennas from vehicle loads, temperature extreme variations, corrosive agents (gas, oil), and water. The integrated encapsulated unit has outer dimensions of 1000 x 500 mm and a maximum height of 8.5 mm, allowing the placement of hard plastic material of 1.5 mm (that is finally applied for integration on the road infrastructure), leading to a total maximum height of 10 mm of the whole strip from the road surface. In addition, its cross-section has a curved shape to minimize

passing vehicles' discomfort. The strips consist of 4 discrete parts (Figure 2.1), hosting the elements described above. Those are installed orthogonally concerning the road lanes' direction, with each lane hosting one strip at the end. A similar but simpler strip configuration has also been created and added to accommodate Vulnerable Road Users (VRU) scenarios. A particular strip is designed and installed on the road pavement that precedes the zebra crossing to detect pedestrians.

2.1.2 Equipped and non-equipped vehicles

The proposed solution aims to constitute a low-cost solution that will host the critical intelligence and source of safety-critical information on the road infrastructure directly and not on peripheral to the road infrastructure elements (e.g., cameras and other surveillance systems) exactly to fill in the gaps if not replace and increase the reliability of the information that is provided already by the latest as well as the onboard solutions. The proposed solution is operating in any environmental condition (rain, fog, snow), which is not the case for the alternative systems in place, while it also provides accurate lane position, which has been one of the fundamental technical objectives of the solution. Collected and processed data are finally communicated to the drivers/riders in the form of ITS messages via ITS-G5 technology (Cohda Wireless MK5-RSU). For the vehicles that are not equipped with the corresponding onboard ITS-G5 module, we developed an efficient message protocol based on MQTT messaging system via LTE. This represents one of the novelties of the project as well since it involves vehicles not necessarily equipped with V2X technology. In the project, therefore, we need to distinguish between two types of vehicles (and users):
Equippend and non-equipped vehicles.

Equipped vehicles

The OBU in the equipped vehicles integrates all the necessary communication modules and software stack for the realization of Vehicle to Everything (V2X) communications and the required interfaces for the interconnection with the vehicles CAN bus system and sensors. The communication between the equipped vehicles and the RSU is performed by means of the V2X technology following the ITS-G5 standard. On the vehicle, as well as the OBU, and HMI is pre-installed on the vehicle standard dashboard. The role of the SAFESTRIP infrastructure for these vehicles is to send messages about the road. The vehicles' state is directly sent from vehicles CAN-bus to the Co-Driver instance. In Figure 2.2 an overview of the communication for the equipped vehicle is shown.

Non-equipped vehicles

While the equipped vehicles receive information onboard, non-equipped vehicles rely on a mobile application purposely developed. The end-user mobile application continuously uploads several important values (i.e., ID, coordinates, speed). Then, the C-ITS-S platform stores the information (for historical purposes) and processes it to finally publish the results by pushing back warnings/ notifications/ recommendations information to the end-user application. This message pushing is made available only to entities that have been assigned to this specific notification service (MQTT) of the C-ITS-S platform. Therefore, each entity (RSU, DSS, HMI, and the Co-Driver explained below) is considered by the system as a software module that communicates with each other via the MQTT protocol. For non-equipped vehicles, the decision-making runs on the C-ITS cloud. Figure 2.2 an overview of the communication for the non-equipped vehicle is shown (together with equipped one).

2.1.3 Communication

For Intelligent Roads, the communication between the road and the vehicles represent the main feature. With such interaction, vehicles drivers (and autonomous vehicle) may adjust their behavior when presented with new information, bringing mobility to a new level of awareness that can be exploited to increase the safety of the roads. The cooperative solution has been implemented to accommodate a series of C-ITS driver/rider applications as well as an application for infrastructure operators that aimed to serve as a proof of concept of the solution and are reflecting the project Use Cases [52]. The sensorial and communication architecture has been designed to provide

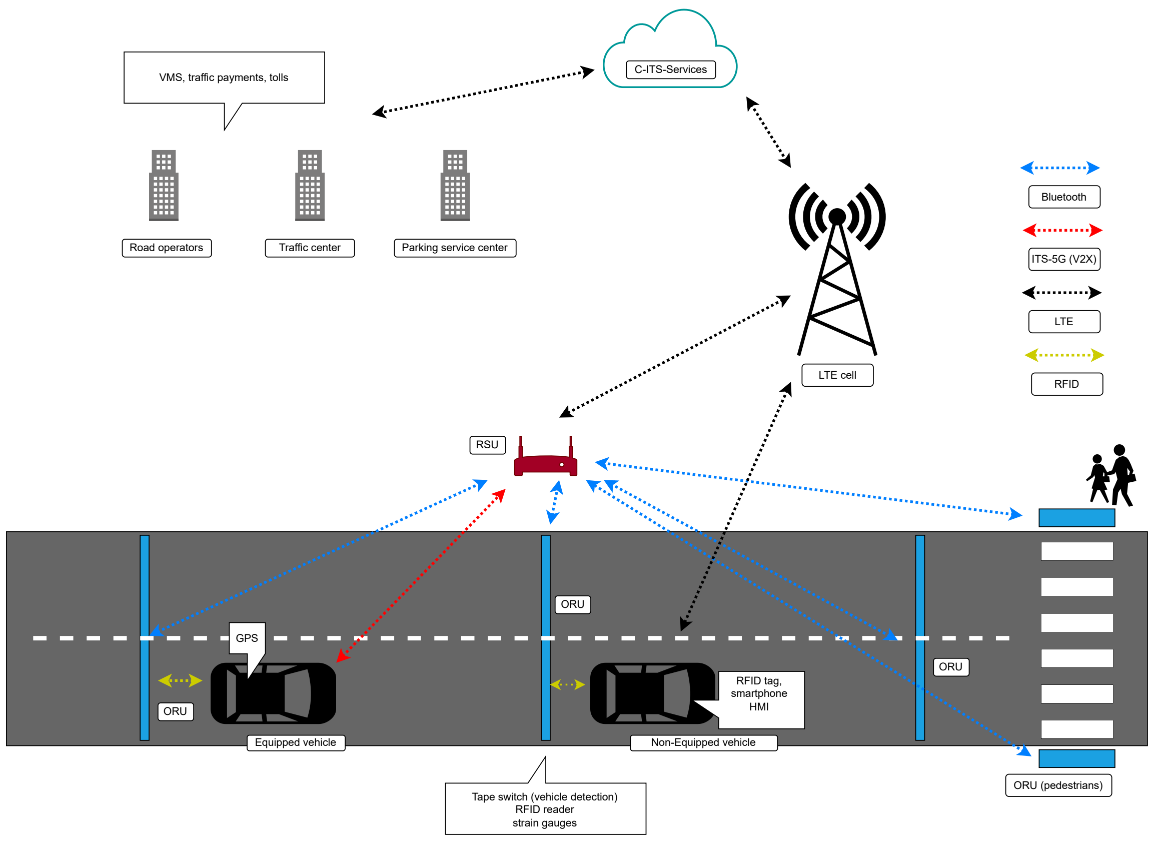


Figure 2.2: Safestrip infrastructure with functional representation of all components.

personalized and accurate information about the vehicles' state and road surface conditions. The information is provided at the lane level with the scope to improve existing safety applications and services or to develop new solutions that were not possible without SAFESTRIP technology. A main objective is to include all other vehicles (the non-equipped) and all potential service providers in the communication and information exchange, and not only the equipped vehicles.

In order to maximize interoperability and transparency to future application developers, The standard V2X is used as reference and the cost to the user is minimized. To this end, the most critical messages defined by the V2X communication (i.e., CAM, DENM, and MAP) is mapped into corresponding messages implemented with the MQTT protocol. The MQTT protocol is selected to communicate with the non-equipped vehicles over LTE 4G channel. The MQTT messages (or topics) are named 'virtual' and are a copy of the original messages of the standard V2X communication as defined by ETSI ITS-G5 or SAE J2735 [104], except for the serialization of the message that is different. With this solution, non-equipped vehicles can receive and send information to connected vehicles and other cloud services using the RSU as a gateway, therefore including, finally, vehicles that are nto connected in term of V2X. Therefore, the equipped vehicles could benefit from applications developed for connected ones, but they can become 'visible' road users to connected vehicles, implicitly improving the safety level provided by the applications.

The application layer does not depend on the type of communication (V2X or "virtual" V2X). The only difference is that for non-equipped vehicles, one instance of the Co-Driver for every vehicle is created in the cloud (or in any server hosting the system) when the smartphone subscribes to the service. The MQTT broker manages messages are a copy of one of the V2X. Therefore, the application does not change. The solution simplifies the application design and maintenance from the developers' perspective since a standard interface can be adopted both for equipped and non-equipped vehicles and for cloud services. Moreover, the same safety application can be hosted in the equipped vehicle or on the cloud to serve non-equipped vehicles, even if with some limitations due to communication delay and lower information update rate. It is important to remark that the "application layer" is not constrained by any communication standard. It can be adapted with the new emerging standard of C-V2X.

2.2 Co-Driver in SAFESTRIP

In this section, we discuss the Co-Driver ADAS applications. Three different applications are addressed. Each of them is related to one or more dangerous situations in mobility. The idea is to apply the concept of the Co-Driver we have seen in Chapter 1 taking advantage of the potentiality of the SAFESTRIP infrastructure in order to give birth to new ADAS systems, especially for the vehicles that are not equipped with efficient onboard sensors and neither with the V2X technology.
At first, in Section 2.2 we discuss a general structure of the Co-Driver that is used among different applications (or use cases), providing a general architecture that is used. The first is about the pedestrian protection at a zebra crossing and it is described in Section 2.3 and indicated with the acronym VRU (Vulnerable Road User), the same application covers another use case, the Wrong Way Driving (WWD). In Section 2.4 an ADAS application that concerns the roadworks on the highway (or other roads) is addressed. The last application is about the intersection in urban and suburban roads (the INT symbol indicates it). This application is the most complex one, and it was conceived to work with more data than the one provided by SAFESTRIP infrastructure.
The safety functions in SAFE STRIP are based on the Co-Driver concept and the idea of the inference of drivers' intentions. The Co-Driver used in this project was built as an extension of the one proposed in [85][34]. The Co-Driver represents a virtual agent used to understand what the driver is doing and provide a warning based on the availability of safe and feasible maneuvers to achieve the goal in the driving scenario (e.g., stopping at a cross-walk). The decision depends on the context and how strong the driver has to react to comply with the safe maneuver computed by the Co-Driver. The Co-Driver identifies the danger, the action required to avoid it, and the driver's effort to change the longitudinal dynamics (i.e., jerk) to enter a safe state. The role of the Co-Driver is also to discriminate when danger is imminent or not. Additionally, the Co-Driver represents a unified approach for all use cases, which allows comparing different hazards with the same metrics rating the most dangerous. This is also the reason why the Co-Driver concept was chosen as background framework for the ADAS application of Sections 2.3,2.4 and 2.5.
Each safety application is built on a **receding horizon approach**. It means that the maneuvers are continuously generated at a fixed rate, calculated by the Co-Driver over a time horizon that moves in front of the vehicle as it travels along the road. The receding horizon schemes give the flexibility to adapt to the driver's change of intention and constantly be updated with new vehicle data or hazards. Every Co-Driver based application structure is composed of two hierarchical layers ("layered control architecture"): the top layer aims at interpreting the environment in terms of dangers (e.g., for instance, "wrong way vehicle" or "pedestrian" in the first use case), and the second is dedicated to the generation of feasible, human-like longitudinal behaviors (or maneuvers) that comply with the above-identified danger. Those longitudinal behaviors are minimum jerk maneuvers that represent a specific action to avoid the danger (for example, the maneuver to stop in Section 1.3.5). These maneuvers are the Longitudinal Motion primitives, and are explained in Section 1.3. The motion primitives are parameterized solutions of optimal control problems that minimize a cost function related to jerk and time. Those solutions resemble longitudinal maneuvers of human drivers, for instance, when they drive from the current state to a complete stop at a pedestrian crossing. In this context, the process of maneuver generation is called "Action priming." These maneuvers resemble the concept of affordances in neuroscience [53] (i.e., maneuvers to reach the desired goals that are also feasible starting from the actual state of the vehicle). The layered control architecture is used to model the human decision process that selects one maneuver among a large set of feasible ones generated using the longitudinal motion primitives (Section 1.3). The block representation of the selected architecture is shown in Figure 2.3. The driver chooses one maneuver, but in our mirroring approach, all the possible maneuvers are taken into consideration to identify the one s/he may use. It is worth noticing that the system does not detect the exact maneuver the driver is pursuing (imitation), but only the compliance with safety is evaluated (interpretation). In Figure 2.3 the general hierarchical architecture is discussed. In general, we decided to use two layers, resembling the layered architecture from Brooks [26]. The Co-Drivers in the SAFESTRIP applications share this two layers architecture. Let us now discuss the single elements.

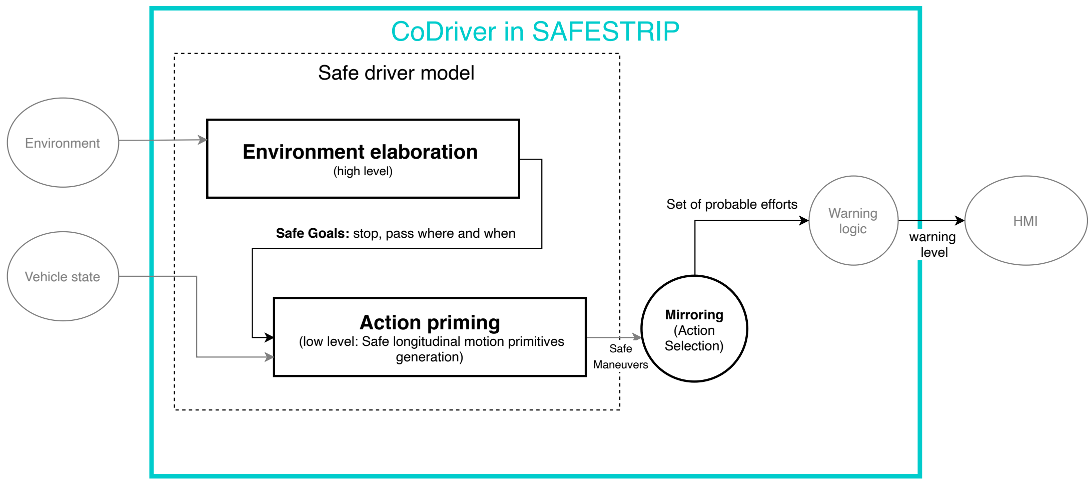


Figure 2.3: general structure of the Co-Driver on which every safety application in SAFESTRIP. The high-level environment elaboration identifies the dangers and provides safe goals to the action priming low-level layer. Such goals can stop before a certain distance or a time slot to safely cross an intersection. The action priming layer is demanded to generate longitudinal motion primitives to accomplish these goals. The mirroring (action selection) block converts these maneuvers into possible efforts, and the warning logic decides if such effort is too severe.

Environment elaboration

This layer generates high-level goals to be executed by the action-priming layer. One goal generated from this layer can be translated into more than one action from the action priming block. In other words, this layer generates affordances' goals like "stop before that distance," but it does not generate the affordances itself. The level of abstraction is higher than the action-priming block. In the Co-Driver instance in SAFESTRIP, the possible outputs of this layer are two:

- **Stop** before a given distance (usually indicated by $s_{f,0}$).
- **Pass** within a given range (or multiple ranges) of time. Indicated by $\mathcal{T}$, that is a union of compact sets, usually no more than two.

The action priming layer elaborates these goals. Let us now make an example of a possible outcome. The first outcome, which we indicate with the keyword "STOP," may occur before a zebra crossing. In figure 2.6, we can see a graphical representation of the interaction between the layers. The environment elaboration layer sends to the action priming block the goal to safely stop. It is worth mentioning that this outcome is not to stop *exactly* at that distance. Instead, it gives just the indication that the driver must stop *before* the distance as a safe behavior.
The environment elaboration layer is also responsible for resolving logical conflicts between different goals. However, this aspect is not discussed here since no experiment was done. Furthermore, such prioritization was based on a very straightforward logic [5].

Action priming layer

The task of this layer is to translate the abstract goals provided by the environment elaboration layer into actual pools of primitives that the driver could pursue. The building blocks of this process are the longitudinal motion primitives (see Section 1.3). This block generates the primitives to plan the motion from the current state to the goals provided by the upper layer. It generates a set of longitudinal motion primitives for each goal. As we have seen in the previous Section 2.2, we have two types of high-level goals, called in short "Pass" and "Stop." The action-priming block receives the necessary data to generate the maneuvers. Let us now discuss the input and the output of this block. Note that more than one goal can arrive from the upper layer, The action priming generates maneuvers for all the goals, assuming they are safe. Let us now discuss how the Action-Priming block elaborates the two types of goals: the Stop and the Pass goal.

The stop goal

This high-level goal comes up in a situation where the only chance to avoid danger is to stop. A typical scenario where this goal is used is the pedestrian crossing, in which one has to stop before the crossing. Therefore, the goal is to provide all the affordances that link the actual states to any distance before a certain distance $s_f < s_{f,0}$ with zero final velocity. The input from the action priming block is, therefore, the distance $s_{f,0}$ (s_f indicates the primitive generation argument, see Section 1.3). The output is a set of primitives of type CSS (see Section 1.3.5), indicated in Equation (2.1).

$$M_{stop}(s_{f,0}) = \mathcal{M}_{\text{CSS}}\left(a_0, v_0, s_f < s_{f,0}\right) \tag{2.1}$$

The set M_{stop} is a continuous set of maneuvers, covering all the stopping positions less than $s_{f,0}$. This set was addressed in Section 1.3.5, in the same Section, in Figure 1.4 left, we illustrated an example of the continuous set of primitives.

The pass goal

The pass goal is more complex than the stop one. It represents the possibility for the vehicle to pass a point within time slots. The point to pass is defined, similarly to the stop case, by a parameter of the distance called s_f. The time slots instead are expressed by a set $\mathcal{T}$, usually consisting in the union of a few compact sets. This set represents when it is safe to cross the point at $s_{f,0}$. The time origin is the current instant since the Co-Driver always operates in a receding horizon approach.

For instance, if the time slots' set is $\mathcal{T} = [5, 7]$, it means it is safe to cross the point $s_{f,0}$ after a time between 5 and 7 seconds.

To accomplish this goal, we need sets of maneuvers that link the current states of the vehicle to a final constrained distance and constrained time, covering all the time slots. The final velocity is instead “free”, given by the optimality of the maneuvers. The chosen type of the primitives is the CST, illustrated in Section 1.3.6, these primitives perfectly fits this type of problem. The set of the maneuvers is given in Equation (2.2)

$$M_{pass} = \mathcal{M}_{\text{CST}}\left(a_0, v_0, s_{f,0}, \mathcal{T}\right) \tag{2.2}$$

In Section 1.3.6 we provide an example of a M_{pass}, with the associated initial jerk set.

To recap the generation of the maneuvers of the two types of goals, in Table 2.1 the input and the outputs of the action-priming block are summarized.

Goal name	Input	Output
Stop	$s_{f,0}$	Set of maneuvers from Equation (2.1)
Pass	$s_{f,0}, \mathcal{T}$	Set of maneuvers from Equation (2.2)

Table 2.1: table of the input and the output of the Action-priming block.

Action selection, mirroring and warning strategy

The outcome of the action-priming block is a set of all the maneuvers. Then, it is passed to the action-selection block (or called mirroring of intention). While the other blocks are demanded to generate goals and actions, the action-selection block must collect the maneuvers and evaluate them on some metrics. It is inspired by the role of the Basal Ganglia [20], in the motion planning of primates.

In this context we are interested in **interpret** the drive intentions. In order to do so, we use an approach similar to [34].

From maneuvers to effort sets: The mirroring process is based on the instantaneous effort required from the maneuvers. Therefore, the first ingredient is a map from the maneuvers' sets to a set of efforts. We assume that the driver controls the vehicle imposing the longitudinal **jerk** $j(t)$ (see Section 1.3). The effort associated with a maneuver is its **initial jerk** $j_0(m)$ or $j_0(M)$, where M is the set of the maneuvers as seen in Section 1.3. A set of maneuvers generates a set of initial jerks. Note that, fixing other parameters conditions, this quantity is monotonic with the decreasing of the distance s_f for the *stop* maneuvers and with T for the *pass* maneuvers. We will use "initial jerk set" and "effort set" interchangeably from now on.

Mirroring as comparison of effort sets: the key idea now is to evaluate if the set of efforts carried out by the

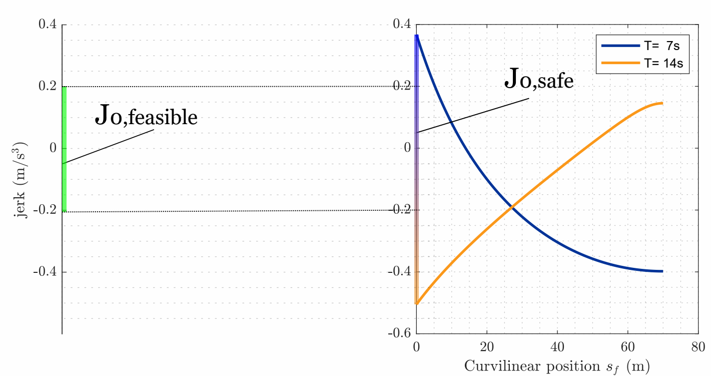


Figure 2.4: Comparison between a jerk set retrieved by the *safe* primitives from the action priming block (right) and the generic *feasible* jerks sets used as a comparison (left). In the case described, the driver is behaving safely since the set of the initial jerks of the allowed primitives overlaps the safe jerks set.

action-priming block (that are associated with safe behaviors) is *feasible*. The objective is to understand if the driver's intention diverges from the safe ones suggested by the previous two blocks. And in order to do so, we want to measure the effort required to follow safe maneuvers from the current state. For example, if the effort is too high for all the safe maneuvers, the driver is doing something different and, therefore, unsafe.
To evaluate this, we need to define some standard efforts set that are "feasible." Once these feasible efforts sets are defined, a comparison is made by simply verifying that the driver safe effort sets and feasible efforts set overlap. Figure 2.4 illustrates a simple example of efforts sets comparison: the green vertical segment indicates a range (sets) of jerks that is considered feasible, the shaded vertical line on the right plot instead indicates the efforts set from the safe primitives. In this case, the sets overlap, and therefore the driver is considered aligned with the safe behaviors. *warning strategy*. We now define the warning strategy, which compares effort sets to generate warning messages of different severity. We have three levels of warning:

Level 0 It is only informatory, the driver is not at risk, but the information on the danger is provided. It is usually associated with the blue color.

Level 1 It is associated with the yellow color, the danger is a real risk, and the driver is demanded to perform correcting maneuvers.

level 2 The red warning, at this level, the driver is demanded immediately to perform the corrective maneuver. The HMI emits a loud sound signal to ensure driving the attention to the warning message.

To understand which warning message to throw, we associate every warning level to a set of feasible efforts set. The wider the set, the higher the warning level associated.
Level 0 is not associated with any set since it is always present if the environment elaboration layer detects a danger.
Level 1 and level 2 are associated with a set of safe initial jerks each. Let us call them $\mathcal{J}_{w,1}$ and $\mathcal{J}_{w,2}$ respectively. As soon as there is an overlap between the initial jerk set of the agent safe primitives (J_0) and the feasible jerks set of the corresponding warning, the corresponding warning is not issued.
Warning of level i is issued if:

$$J_0 \cup \mathcal{J}_{w,i} = \emptyset \tag{2.3}$$

Note that to ensure the warning levels to be progressive (red 2 implies yellow 1 that implies blue 0) the set must include each other in sequence:

$$\mathcal{J}_{w,i} \subset \mathcal{J}_{w,j} \quad \forall i < j \tag{2.4}$$

As already stated, the blue warning essentially is associated with an empty set being issued in any case. In this work, the warnings jerk sets are symmetric to the origin and read as follows:

$$\begin{aligned} \mathcal{J}_{w,0} &= \emptyset \\ \mathcal{J}_{w,1} &= [\,-1\,,\,1\,] \quad (ms^{-3}) \\ \mathcal{J}_{w,2} &= [\,-3\,,\,3\,] \quad (ms^{-3}). \end{aligned} \tag{2.5}$$

In Figure 2.5 several examples are illustrated to clarify how the warning strategy works.

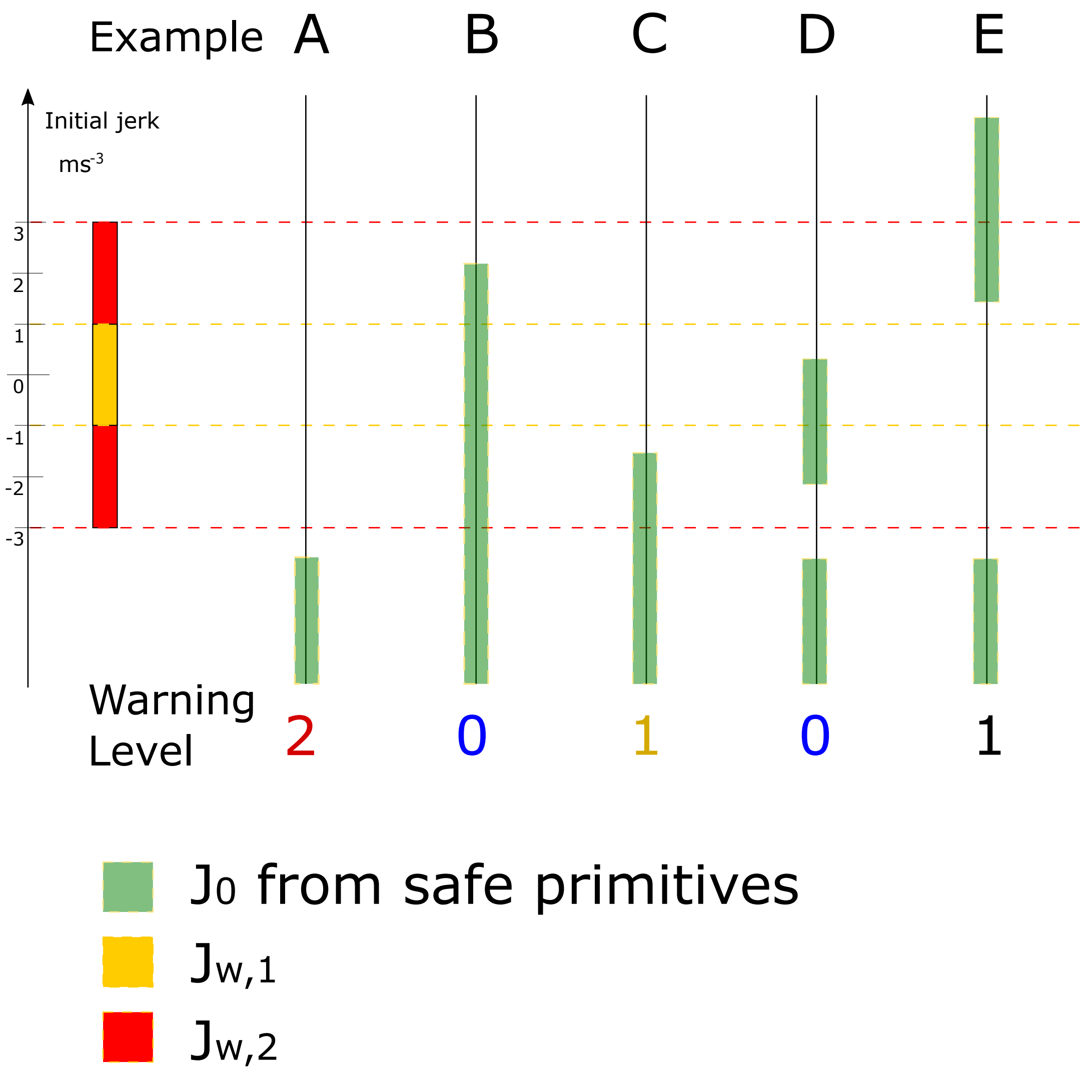


Figure 2.5: Five examples of comparison between efforts sets (J_0), and warning jerks sets ($J_{w,i}$). In the first case, case A, the efforts set do not overlap with any of the warning sets. This leads to warning level 2, which is the maximum. Case B instead presents efforts set that overlaps with $J_{w,1}$, only the blue warning is issued. In the third case (C), J_0 overlaps with the red set but not with the yellow set; this means a warning of level 1 is issued. The other two cases (D and E) replicate cases with warning levels 0 and 1, but the effort sets J_0 are discontinuous, showing that what matters is if the set overlaps or not with the warnings feasible jerks sets.

For every application of Co-Driver in SAFESTRIP, these three blocks are defined. An example of how the Co-Driver works, block by block, is illustrated in Figure 2.6.

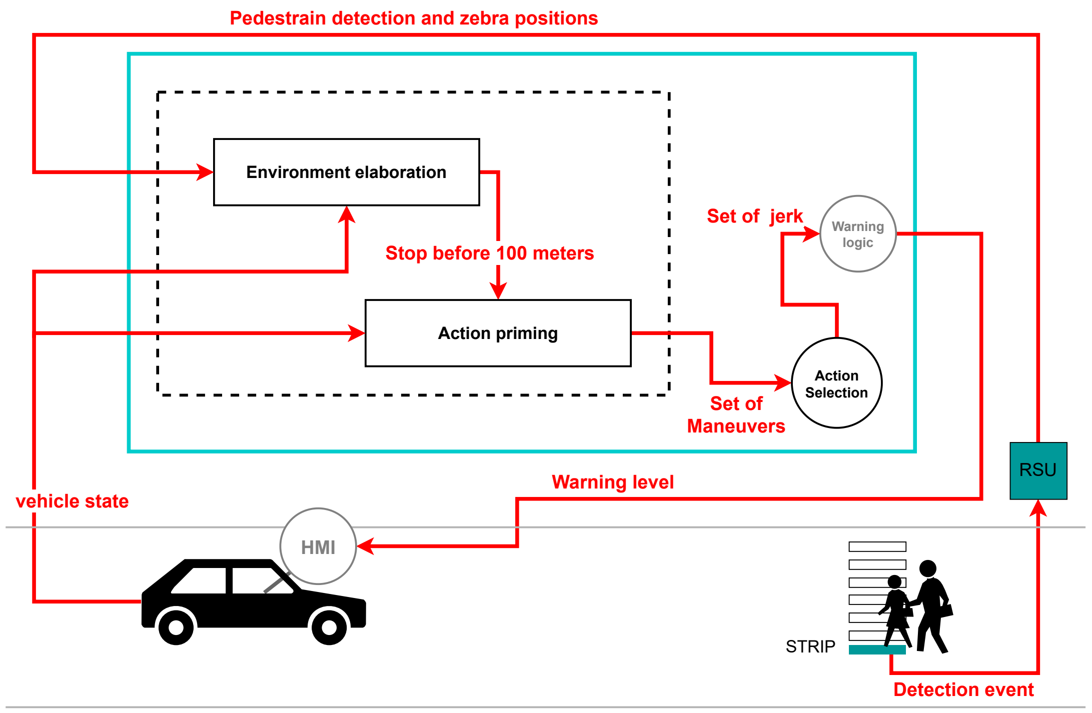


Figure 2.6: Example of how the Co-Driver works in the pedestrian use case (see Section 2.3). The SAFESTRIP system detects the pedestrian. The environment elaboration layer receives this message and sends to the Action priming layer the goal to stop before the pedestrian position (100 meters in the example). The action priming layer uses the vehicle states to generate the set of safe primitives, like the one we have seen in Section 1.3.5, in Figure 1.4. An initial jerks (efforts) set is associated with the primitive's sets by the action selection layer. That effort set is then used by the warning logic to throw a warning if needed, as shown in Figure 2.5.

2.3 Vulnerable road users and wrong-way driving

Two use cases are condensed into one safety application. These use cases concern the presence of pedestrians on a zebra crossing. This use case can be potentially extended for all the hot spots where vulnerable road users and vehicles like cars interact. These two use cases are addressed together since the adopted solution is very similar, and also, in the SAFESTRIP project, they were addressed together under the name of "cooperative safety functions." Thus, the first scenario considers the collision between pedestrians and vehicles, which is known to cause a high number of accidents with a high rate of fatalities (21% of all fatalities in EU in 2016 [7]) and serious injuries. The second scenario involves vehicles traveling in the wrong direction from a highway ramp (i.e., out of a service station). These situations, despite being relatively limited in number, have much more severe consequences than those of other motorway injury accidents [6]. The applications are designed for vehicles connected or equipped with sensors to improve the performance of existing similar functions exploiting the cooperative capability of the V2X technology and the information provided by the intelligent strips introduced by SAFE STRIP. On the other hand, the goal is also to have the same applications available for non-equipped vehicles thanks to the particular communication protocol introduced by SAFE STRIP that maps CAM, DENM, and MAP messages of V2X ETSI standard [46] in the LTE 4G channel. This parallel architecture of communication allows non-connected vehicles (called Non-equipped in this project) to receive the warnings and information for each specific scenario from the applications running in a cloud server just by simply installing the SAFE STRIP application on the mobile phone or, potentially, on the on-board infotainment system as Android Auto and Apple Play. The underlying idea is to augment the penetration rate of cooperative safety functions in the market, improving the overall safety on roads. Additionally, SAFE STRIP applications aim to increase acceptability by providing timely, consistent, and personalized warnings. The acceptability is improved on one side using the CoDriver concept that evaluates the driver intention and generates a warning consistent with the danger and the estimated intention and the drivers' specific reaction time. Further personalization is obtained displaying messages only when necessary and relevant to the specific vehicle, which is possible since SAFE STRIP also provides the lane of the vehicle is driving in.

2.3.1 Overview of existing solutions

A large portion of the literature and the state-of-the-art applications in pedestrian protection rely on on-board sensors to detect the VRU, as it is reported in one of the latest surveys on safety issues of vehicles of the future [15]. However, on-board sensors need a line of sight to detect the pedestrian, which is not always the case in a complex urban environment (e.g., because of a parked car or a stopped bus). To overcome this problem, different solutions have been proposed. A new trend in the field consists of cooperative sensing that shares information produced by on-boards sensors of one vehicle to others connected to it. In this way, each vehicle extends the horizon covered by its sensors (for instance, limited by the field of view of the cameras, radars, and lidars) in a cooperative manner using the information perceived by other connected vehicles in the surroundings. Additionally, the position and speed information gathered by the sensors embedded in the mobile phones carried by the pedestrians or cyclists can be included in the same cooperative framework. For Example, in [71] the mobile phone is used to detect the presence of a pedestrian close to the road and estimate his/her trajectory in order to understand whether he/she has the intention to cross the road. The information is then delivered via 4G/LTE and DSRC (ITS-G5 standard) communication channels to connected vehicles. There are also IoT solutions based on wearable devices as in [120] where the system was used to detect a distracted pedestrian. However, the reader may note that the direct communication between the car or the infrastructure and the pedestrians' smartphones or wearable devices can be jeopardized by the latency of the communication and the inaccuracy of the GPS positioning embedded in the device. As a third family of methods, VRU can be identified by sensors (e.g., camera or lidars) integrated into the infrastructure on high poles to cover as much as possible the area of interests [15][44][56]. The latest solution was investigated at various levels, including European projects, and it is capable of spotting the presence of vehicles, pedestrians, or cyclists (even non-equipped vehicles) via classification algorithms, also with some partial occlusion if powered by Deep Learning [44]. Nevertheless, such a solution still needs a good line of sight and cannot work with heavy rain, snow, and fog. In conclusion, we can say that the vulnerable road users' biggest issue lies in their detection. The proposed solutions strongly depend on on-board sensors, or sensors carried by the pedestrian when they are not embedded in the infrastructure. This aspect reveals the need to have a robust solution that is not dependent on driver's car equipment or the pedestrian choice to use portable devices. The new solution should also cover the limitations (mainly occlusions and weather conditions) of the detection systems based on video cameras, lidar, or radars installed in the infrastructure. On the other hand, wrong-way driving detection is treated separately in the literature and has been less investigated than VRU protection. Nevertheless, it is a dangerous scenario that, when it occurs, causes a high rate of casualties [131] [59]. Among the solution proposed, passive detection through infrastructure sensors is by far the most widespread approach that employs sensors such as conduction loops embedded in the road surface, microwave doppler radar, ultrasonic sensor, photoelectric sensor, and video image processing technology [131][72]. Image processing and photoelectric sensors successfully prevent wrong-way driving, but a careful installation to optimize the detection angle is required. It has also been proposed to use the attenuation induced by a passing vehicle on the signal received by a cluster of radio modules placed on the roadside [59]. The system can identify the position, speed, size, and direction of travel of the vehicle producing a local warning to the driver and sharing the information to the other road users in the vicinity but with a delay time of about 6s, which can be too long for motorway scenarios. SAFE STRIP comes in to fill this gap with a solution that is entirely independent of on-board sensors or sensors carried by the VRU, and it can detect the wrong way driving vehicles or the pedestrians where they are, in all weather conditions without the need of a line of sight and at a lower cost.

2.3.2 Pedestrian (VRU) use case

The purpose of the system is to improve the safety of pedestrians when they are crossing the road. The system is then conceived to protect them where they are allowed (and they have the right) to cross the road in complete safety.
therefore, the system is conceived for the Zebra crossing. Since it requires infrastructure on the road, it must be used in precise spots. The strips, in particular, must be placed in the vicinity of the zebra crossing, where the pedestrians are supposed to pass while approaching the cross-road. If the surface of the strip is enough, the detection of the pedestrian is reliable (according to the reliability of the pressure sensor used for this purpose).
If pedestrians use the zebra crossing, they do not need any equipment. The system relies on only on-road sensors. The use cases involve any pedestrian that is about to cross the road. It is not even necessary that the pedestrian is aware of the system. The system acts on the drivers driving their attention.
The driver's attention to the road markings depends on several factors. Therefore, the pedestrian protection system must compensate for these factors. They can be internal (the driver is distracted) or external. The external causes are the biggest motivation to develop the system since they affect every driver. Such causes are related to visibility since several factors can limit it. They can be environmental, like the fog or the poor illumination in the night, or occlusions in the field of view: a parked vehicle or a vehicle that already stopped at the crossing. The system does not care about what is causing the driver's inattention. The Co-Driver can estimate the driver's intention relying on the vehicle state only and therefore does not need to know what is causing the possible lack of attention, covering internal and external causes. Based on this assumption, we can imagine several situations in which the VRU protection is suitable. For instance, a pedestrian is crossing the road, and the driver is distracted, or occlusion is present, and the driver does not see either the zebra crossing or the pedestrian being tempted to overtake the occlusion. This application will also cover low visibility conditions that cause the driver not to see the pedestrian in time to stop. Such situations are depicted in Figure 2.7. The system can operate with more than one driver simultaneously since the only interaction taken into account is the one between the driver and the pedestrian.

2.3.3 Wrong-way driving (WWD) use case

This use case regards the danger of crashing against a vehicle driving in the wrong way. The application uses the SAFE STRIP infrastructure to detect a vehicle using a wrong motorway exit or a rest area, leading to a dangerous situation. The vehicles following the normal traffic flow are warned similarly to the VRU use case and the wrong way to drive a vehicle if equipped or with SAFE STRIP application. The only difference lies in the type of danger indicated in the HMI. The point of interest about the danger is the point of a possible collision, which in the case of the ramp is its starting point (the merging area with the traffic). When a car has passed the dangerous point, or the danger has disappeared, the warning is switched off.

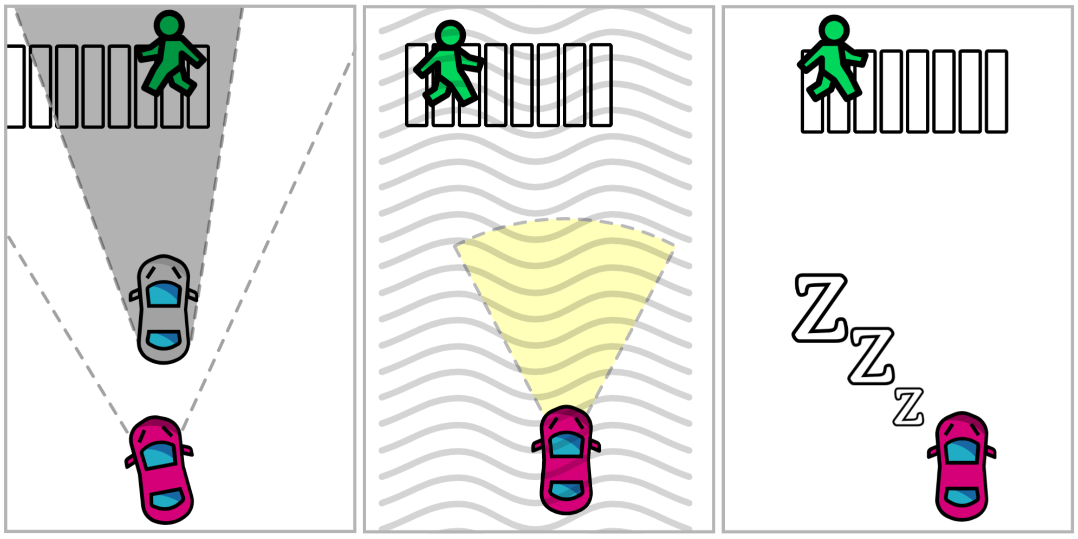


Figure 2.7: Example of situations covered by the VRU use case: occlusions, low visibility, and distraction.

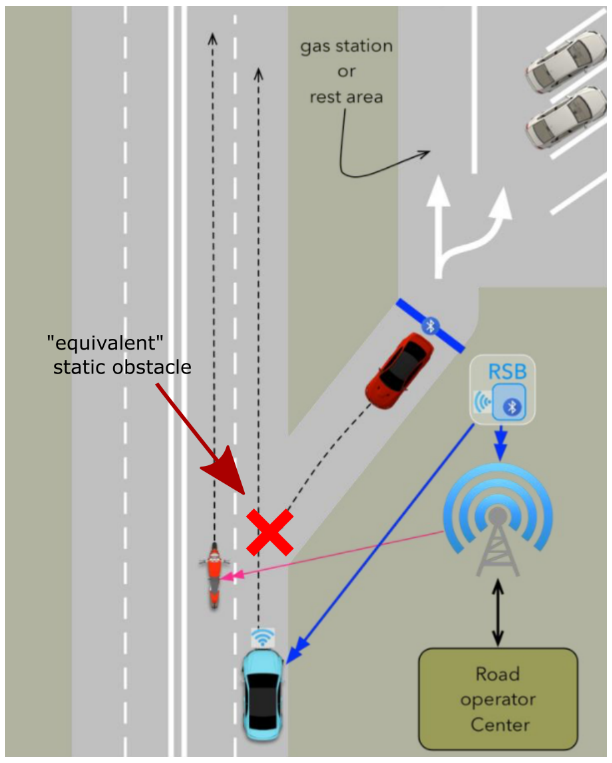


Figure 2.8: Wrong Way Driving (WWD) use case: vehicle approaching in the wrong way on a ramp is considered as a "static" obstacle but positioned at the beginning of the ramp, considering the worst case in which the vehicle reach the end of the ramp entering the road.

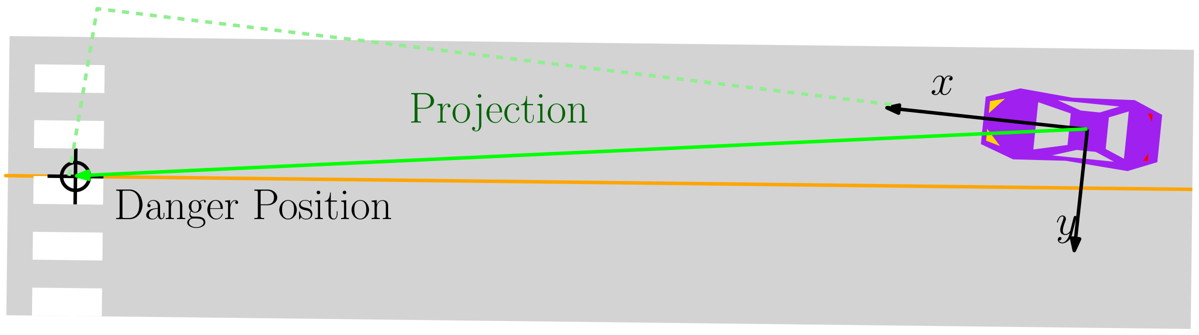


Figure 2.9: Position of the danger projected in the vehicle reference frame. For non-equipped vehicles, the heading is approximated by the orientation of the strip.

2.3.4 Co-Driver instance

As stated at the beginning of this Section, every application is based on a specific instance of the Co-Driver, based on the general scheme of Section 2.2. In order to identify the Co-Driver instance, we need to define the three main blocks: Environment elaboration, Action Priming, and Action selection (mirroring).

Environment elaboration

This block performs a straightforward task in terms of logic (w.r.t the other applications). First, it receives a message from the SAFETSTRIP infrastructure containing the danger type and the position. Then, the first operation performed by the loop is to project the DENM position in a reference frame attached to the vehicle (Figure 2.9). This step is performed for every application. This block has to establish if the danger is incoming or not. It is based on two simple criteria: the distance and the heading. If the distance is above a threshold or the vehicle is not heading towards the danger position, the latter is considered "not incoming," and the warning system is not triggered (the Co-Driver).
If the danger is in front of the vehicle, the goal that is returned to the action priming block is to stop before $s_{f,0}$, that is the distance between the vehicle and the danger.
We decided to use the euclidean distance to ease implementation and not a curvilinear distance following the road. In other words, we neglect the curvature of the road between the vehicle and the danger. This approximation is a conservative solution for safety since the euclidean distance is eventually lower than the curvilinear distance. Furthermore, this choice simplifies the application significantly since we do not require road geometry knowledge, facilitating a real-world application and the experiments without affecting the safety. Therefore, the goal carried out by this block is to stop only. One may argue that when you wait for a pedestrian to cross the road, you do not have to stop all the time. Sometimes it is enough to slow down, and it is indeed true. Our Co-Driver understands this behavior since you are always ready to stop if needed when you slow down. The Co-Driver is designed to evaluate the pool of potential intentions, and in the case above, you keep the possibility to stop in your pool of feasible maneuvers, and the Co-Driver detects that.

Action priming

Given the goals provided by the higher layer, this block is ready to generate safe primitives. The set of safe primitives is computed at this point: the vehicle's state is used for the initial conditions of the maneuvers. Before calculating the maneuvers, the reaction time is taken into account, propagating the initial conditions and the distance with constant acceleration. New initial conditions are then defined according to the time reaction compensation. For simplicity of notation we consider $[a_0, v_0, s_{f,0}]$ as already compensated with the reaction times. The manoeuvres set M_{stop} is then calculated as a set of CSS maneuvers (See Section 1.3 for the notation):

$$M_{stop} = \mathcal{M}_{\text{CSS}}\left(a_0, v_0, s_f < s_{f,0}\right) \tag{2.6}$$

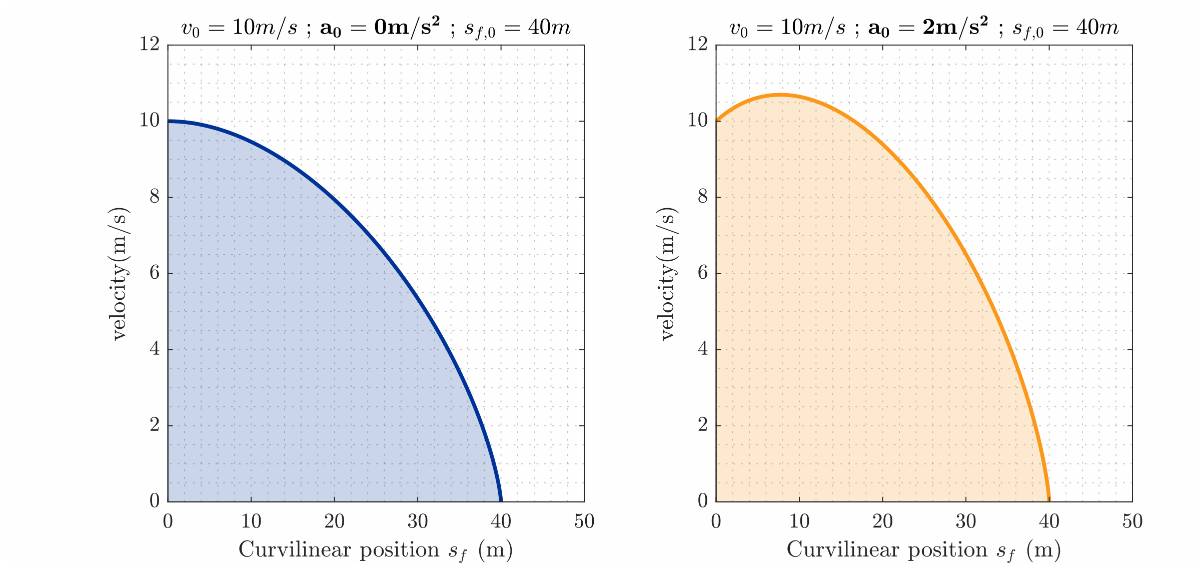


Figure 2.10: Example of action priming for stop maneuvers in VRU, the safe maneuvers are all the maneuvers to stop before the position $s_{f,0}$. The two cases differs only for the initial acceleration a_0.

The set of maneuvers M_{stop} represents all the possible maneuvers that have initial conditions a_0, v_0 and stops before $s_{f,0}$. Note that the lower the distance, the higher the deceleration and the initial jerk.
The initial jerk is evaluated to measure the driver's effort to change the speed profile and perform the stop maneuver. As stated in Section 1.3.5 the higher the magnitude of this value, the higher the effort. Intuitively, the jerk is proportional to the rate of change of the brake pedal position. The efforts set reads as:

$$J_0 = [-\infty, j_0 \left(\mathcal{M}_{CSS} \left(a_0, v_0, s_f = s_{f,0}\right)\right)] \tag{2.7}$$

We consider the jerk of the slowest maneuver as the maximum (minimum in magnitude), and we can use it directly to compare it to the thresholds. Two examples of the action priming results are illustrated in Figure 2.10, we vary the acceleration since it's effect is not immediate to visualize as for v_0 and final distance $s_{f,0}$. The filled area reresents all the maneuvers in the set. We are not interested at this point in evaluating single maneuvers. Instead, we are interested in the "efforts set," the set of the initial jerks J_0. The maneuver with the higher distance is the one with the higher jerk. A note that the j_0 is taken with sign.

Mirroring of intention

As discussed in 2.2, the mirroring of intention is about the initial jerk set J_0. We compare the set J_0 with sample sets of jerks for yellow and red warning levels. The general reasoning reduces in a comparison between $j_0 \left(\mathcal{M}_{CSS} \left(a_0, v_0, s_f = s_{f,0}\right)\right)$ (j_0 for brevity) and two thresholds, as discussed in Section 1.3. The maneuver should be negative during braking to increase the deceleration (even positive for some maneuvers if the driver is decelerating more than necessary). Therefore, lower jerks (i.e., more negative) require more effort to brake. For this reason, to take into account the effort of the maneuvers, one must take into account the jerk with sign. In these applications, we always consider symmetric thresholds, but one can potentially use a threshold for negative and positive jerks.
The warning system is detailed in Section 2.2, as well the values used. The action selection here, in presence of only stop primitives, reduces to the comparison between j_0 of the slowest primitive and the negative threesholds of our feasible efforts sets.
FInally, no warning is issued if j_0 falls above the threshold corresponding to warning level 1 (yellow band in Figure 2.11). If j_0 is below the threshold of level 2, warning level 1 is issued otherwise, the warning of level 2 is issued. The purpose of this mirroring process is to issue the warning when it is essential and to deactivate it (warning level 0) when the driver is aware of the danger, and s/he is reacting appropriately (i.e., changing the

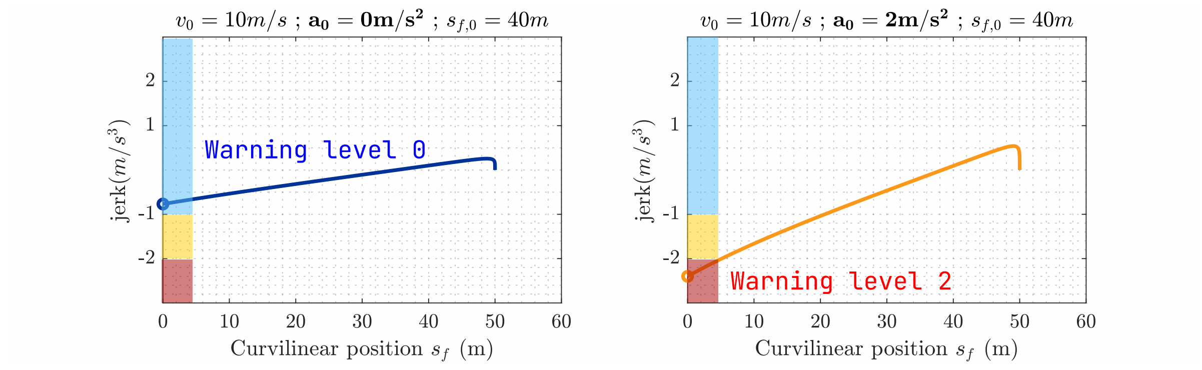


Figure 2.11: Action selection for stop maneuver in the VRU and WWD danger cases, in the shown situations, the system issues warning level 0 and level 2, respectively. Note that when the set is composed only by stop maneuvers, it is a lower unbounded set; therefore, we do not consider the positive thresholds. We only need to compare j_0 with $j_{w,min,1}$ and $j_{w,min,2}$.

deceleration according to the computed safe stop maneuver).

Results examples

While the general results are shown in Section 2.6.1, we hereby provide an example picked from the equipped vehicles experiments (for the VRU use case). In this case, we choose to show the equipped case since there is the possibility to visualize the velocity trend in time (for each step every 50 ms). Let us now recap the level of warnings and the associated colors: Level 0 warning is blue, and it is only informative, Level 1 is the Medium severity warning, and it is yellow, Level 2 is the highest severity warning, and if issued, means that the required maneuvers are considered emergency maneuver, the color is red.
Figure 2.12 shows in the top part the speed profiles of three tests where the car was driven towards the crosswalk (on the left) at different velocities. The speed profile lines are colored with four colors depending on the warning issued. The blue, yellow, and red colors correspond to the warning level described above. The green color corresponds to no-warnings. The green color is the dominant part of the maneuver's initial phase since the vehicle is far away from the crosswalk (danger is considered not incoming). When the vehicle passes over the first strip, the system is activated, and the blue color associated with level 0 is issued if a feasible and comfortable stop maneuver is still available. Then the three drivers receive a different sequence of warnings based on the evolution of the vehicle state (i.e., speed and acceleration and distance to the crosswalk). In two cases, the drivers receive the yellow warning, and he/she adjusts the velocity to stop safely. In one case, the driver action is faster (i.e., higher initial deceleration phase), and the warning disappears earlier (e.g., the case with higher top speed). The critical thing to notice is that the warning comes back to zero when the driver intends to stop and decelerate properly. The blue color visualizes this following the yellow one when the driver slows down enough, and the corresponding initial jerk falls below the threshold, which means there is no need to change the deceleration. The action taken will guide the vehicle to the desired stopping point, the danger may also disappear before the maneuver is finished, and therefore the driver is not demanded to stop if not needed. What is ensured is that the driver is *ready* to stop safely if required. The result in 2.6.1 proves the capability of the Co-Driver to infer the intention of the driver/rider and remove the warning when the human is doing the right action. In other words, the Co-Driver has understood that the driver/rider has taken the correct action to stop safely. In the third example, the driver did not slow down the vehicle enough, and the warning rapidly changed from level 1 to level 2. In this case, the driver is particularly aggressive, and it does not stop. The initial lower velocity and a slight deceleration profile caused the warning to trigger later than the other two cases because the driver's intention at the beginning is correct.

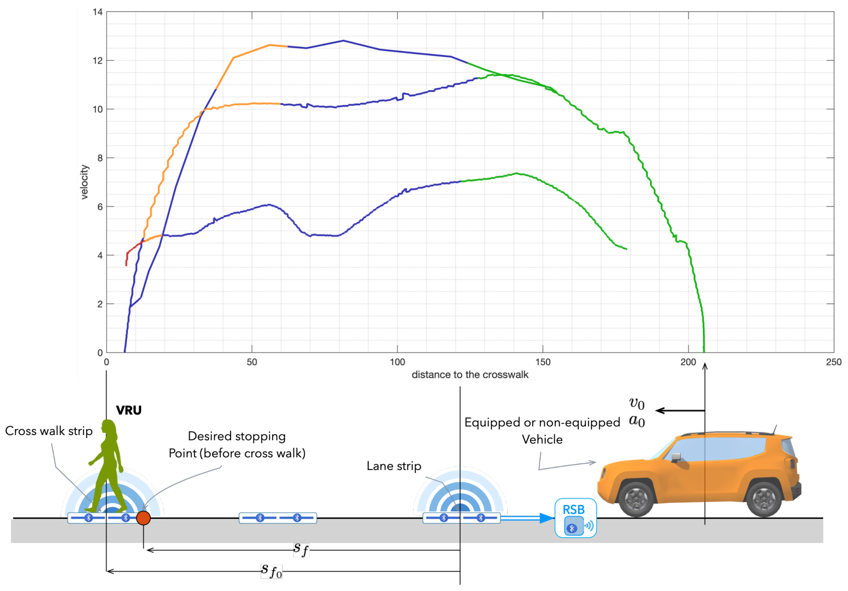


Figure 2.12: Example of VRU application in three different cases. The example are picked to show how the different levels of warnings are issued.

2.4 Roadworks safety function

As stated earlier, this use case concerns the work zones on the highways (or even urban roads). We consider the danger introduced by the closed lanes for the roadworks that often lead to rear-end collisions, dangerous maneuvers in general due to the "too late" lane change, and even invasions of the work zones that may lead to injuries to the workers.

2.4.1 Overview of existing solutions

Another critical point for road safety due to infrastructure is to anticipate to the driver temporary maintenance interventions such as work zones (roadworks). Roadworks are a safety concern, particularly at the beginning of work zones and carriageway changeover points where inattention mainly causes rear-end accidents [12]. In these critical zones, drivers at risk and road workers find themselves in dangerous situations that could potentially be struck by motor vehicles. For instance, according to the National Safety Council, over 100 road construction workers are killed in construction zones on an annual basis, and nearly half of these are killed struck by motor vehicles [31]. Similar figures can be found in [45]. Additionally, roadworks also have a substantial impact on traffic flow, significantly affected by how physical layout changes, as well as the length of the roadwork site, citebakaba2016rdw. A problem that may arise in work zones may be signalized differently depending on the country and the type of roadworks, making it difficult for on-board sensors to detect the zone and the road layout change. Furthermore, most running vehicles are non-equipped without any of these sensing capabilities. Another common aspect is making decisions considerably in advance (like changing lanes for highway work zones). This makes again the possibility to use long-range communication a desirable solution. It can even improve the effect on traffic in that case, avoiding dangerous and delayed lane changes maneuvers by managing the traffic flow in advance to reduce queue. In a recent review on work zones safety systems [31], the authors identify three categories of work zone accident prevention systems. The first is called Speed reduction systems, that focus, as the name suggests, on reducing the motorist's travel speed with or without physical impact on the vehicle. According to the review, these methods appear to be the most effective, as long as the speed reduction is consistent. To achieve this speed reduction, enforcement measures are needed. The second one aims to directly avoid the collision with the work zones equipment by warnings sent to the driver (and eventually to the workers), which was named Intrusion prevention and warning systems. The third category is called Human-machine-interaction detection systems, and it is focused on the accidents inside the work zones, which try to prevent accidents caused by the improper use (or interaction) between construction vehicles and workers. In this context, an established cluster

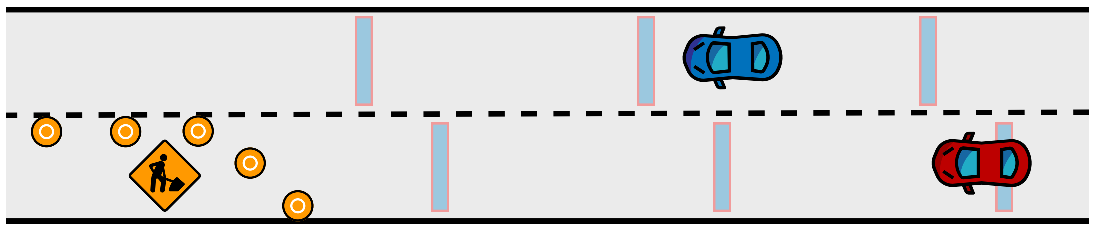


Figure 2.13: Roadworks use case, one or more lanes are closed, the Strips are placed on the road, for each lane.

of safety systems is one of the Intrusion Alert Technologies (IAT) [11]. The scope of these technologies is to advise the workers of a vehicle that is going to collide or intrude into the work zones. The common characteristic of these approaches is using a variety of sensors in the work zone. The sensors technologies are infrared beams, microwaves, and pneumatic pressured tubes as triggering. More recently, IATs have also been equipped with Intellicone, Sonoblaster, and advanced warning and risk evasion (AWARE). A unified study on these methods in [11] provides the primary limits on their effectiveness. In particular, they do not provide sufficient reaction time to the workers (most of these approaches focus on proving advice to the workers and not the drivers). A consistent limitation is also related to false alarms; this will undoubtedly affect the system's acceptance. In the review of [12] it is discussed that a primary focus of the technology is to design blocking devices (like barriers, shadow vehicles, guardrails). These methods are undoubtedly effective at the very end for the workers, but they may not be very effective for the drivers whose vehicles are physically stopped, even minimizing the damage. DRIVE C2X project proposed a solution that enables road operators to communicate information about road works and restrictions to drivers [112]. This solution is implemented with "road-side units," mounted at road works locations, enabling messages and instructions to be sent to approaching drivers directly via ITS-G5 communications or the cellular network. This service applies to all road and vehicle types, but, as far as it is understood, the communication is a static warning sent to all vehicles approaching the area. Furthermore, no personalization is considered, for example, considering the lane the vehicle is driving and its actual speed. The SAFESTRIP solution is oriented to inform the driver in advance of the road works and personalize the warnings, minimizing false-positive warnings thanks to the Co-Driver framework and knowledge of the lane the vehicle is driving in (see section 3.2). On the other hand, according to the review, C-ITS is not mainly exploited in work zones use cases. This makes the system classified between the first and the second category because it implies detecting single vehicles (like the second category, the IATs), but it also communicates to the drivers to change their behavior, like a speed reduction system does. A similar approach is also applied to the unprotected railway level crossing in cooperation with the project SAFER-LC [105] that communicates the approaching train position and speed. SAFE STRIP complements the V2X safety application developed in the SAFER-LC project broadcasting the information about the train to non-equipped vehicles via the LTE 4G communication channel provided by SAFE STRIP.

2.4.2 Roadworks use case

This use case significantly benefits the SAFE STRIP infrastructure since it extensively uses lane information. When work zones are present, some lanes are closed while others let the traffic flow. Therefore, the lane level information, and the knowledge of how the road layout has changed, is of paramount importance for the safety application to decide which specific vehicle sends the warning. Figure 1 shows the work zone scenario layout with one lane closed and the other used to let the traffic flow. Two vehicles, one per lane, are considered for illustrative purposes, but it is not a limitation of the application that can scale to a more significant number of vehicles and several lanes. (The layout presented in Figure 2.13 is also the layout of the experiments).
The danger associated with the work zones depends on whether the vehicle is traveling in the closed lane or not or on the presence of a queue in its lane. Accurate lane identification cannot be achieved using standard GNSS or GPS since the minimum uncertainty on the vehicle position obtained with GPS is still in the order of magnitude of the lane width [7]. On the contrary, the SAFESTRIP system can locate the vehicle's traveling lane. Additionally, this use case can be extended for any roadworks type. The only requirement is to distinguish between closed and

open lanes. The strips can also be placed on custom lanes introduced by the roadworks to create safe corridors for the vehicles, rendering SAFESTRIP solution a flexible approach for temporary modification of road layout. They would be identified as "open lanes" and all as closed.

2.4.3 Co-Driver instance

As stated earlier, the Co-Driver agent is a bio-inspired framework used to estimate the driver indented maneuver about the potential danger ahead. This includes the roadworks, the low-level part of the agent that ultimately the risk level is the same as we have seen in 2.3. At first, the action priming generates all the feasible maneuvers compatible to avoid the potential danger, and we will see that we use only stop maneuvers. Secondly, action selection rates the generated maneuvers based on the effort required and, if the best is not within the threesholds generates the appropriate level of warning. Finally, the action priming block is fed by the information on the danger from the environment elaboration block that interprets the scenario through the V2X (and MQTT equivalent) messages, in which there are the differences w.r.t the application that we have seen in 2.3. The process is continuously repeated to adapt to the changes performed by the driver, and a warning level (i.e., informative, medium, and high severity) is displayed to the driver when he/she does not react appropriately. The Co-Driver and the mirroring of intentions concepts are used to design applications aiming to maximize user acceptance. The concepts and the basic working principles are described in more detail in 2.2.

Environment elaboration

Again we have a Co-Driver which is composed of two hierarchical layers. The outer layer aims at interpreting the environment and extracting safe objectives to follow (e.g., stop before the closed lane). Essentially its role is to interpret the information received about the road scenario and identify the type of maneuvers that the driver can execute. Those high-level maneuvers for road works can be:

- Stopping before the danger
- Going along the lane
- Change lane

The inner layer generates a large set of primitives for these goals. The first intention is modeled as the classic stop maneuver we have seen. The second is present only if the lane is open. The change lane intention cannot be represented by a longitudinal primitive, but we use the stopping intention to measure the safety of the potential lane change maneuvers.
The scheme in Figure 2.3 represents the general layered architecture. In a roadworks scenario, the danger occurs in specific locations and can be immediately defined without the need to resolve conflicts, but in this case, the danger also depends on the lane the ego vehicle is traveling. Therefore, the safe goals of the corresponding maneuver the driver should follow can be defined based on the vehicle location, lane, state, and distance to the danger.
The danger state depends on the lane the vehicle is traveling in. If the lane is closed, the safe maneuvers correspond to any maneuver (possible limits on velocity are not handled here for simplicity reason, but integrating some messages for them would be straightforward, even in the Co-Driver paradigm). These lanes can keep on traveling at the desired speed. There is also another possibility that corresponds to a vehicle that changes to the free lane. We remark that the outer loop must know the same lane the vehicle is traveling in to compute the different maneuvers, and the strips provide this information. This also represents an innovation for equipped vehicles, which GPS cannot identify the lane with the same accuracy as the strips. Despite that the safe goals are immediately defined, it is not straightforward how a human would pursue that goal. To understand if the driver's goal (i.e., his/her intention) is safe, the Co-Driver must mimic how a human driver will execute the maneuver controlling the longitudinal and lateral dynamics of the vehicle. It is worth noting that most of the maneuvers feasible in the targeted scenarios are pure longitudinal maneuvers. Therefore, to mimic how a human would drive to comply with the goal, the aim is to predict the speed profile's future evolution based on the vehicle's actual state and a final safe state (i.e., the goal). Among all possible ways to execute the maneuvers, the one that minimizes the effort is preferable and closer to the driver's choice. Using a simple longitudinal dynamic model, the problem can be formulated as finding the maneuver that connects the actual vehicle state and a desired final goal, or state, minimizing the longitudinal jerk (i.e., derivative of acceleration). The resulting maneuver is called longitudinal

motion primitive and, by definition, is a minimum effort and safe maneuver since it respects the imposed goal, which derives from safety criteria (e.g., stopping before the danger). In Section 1.3 the longitudinal motion primitives are detailed. The jerk minimization yields smooth speed profiles that resemble those produced by a vehicle driven by humans. The optimal criterion finds its theoretical basis in the neuroscience field where the minimization of post-movement errors and rejection of neural noise translates into the minimization of the control variable, in this case, the jerk (See Chapter 1). The shorter the distance, the more the longitudinal acceleration has to be changed to reduce the speed in the given space. Being the jerk, the rate of change of the acceleration, its initial value measures how much the driver has to modify the maneuver to stop the vehicle. As we have already seen. Therefore, the initial jerk measures the driver's effort in pursuing the maneuver.
For a vehicle in a closed lane, there are only two ways of escaping the situation: the first is to stop before the end of the lane, the second is to change lane sufficiently before the end of the lane. We cannot generate lateral maneuvers, our Co-Driver is based only on longitudinal inputs by design, but we unify these two safe maneuvers set into one. We do not directly evaluate lane-change maneuvers. Instead, we evaluate the timing a lane change must be performed. In practice, we answer the question, "how safe is a time the driver starts to change lane is a lane is closed?". The first answer to the question may be "far enough from the end of the closed lane" to quantify this distance from the end of the lane; we use the same metric of the stop maneuvers. Long story short, safety is evaluated as "the driver should change lane before stopping is too dangerous." Note that being the metric the same, if the driver slows down properly to stop, he/she will get no warnings, as well as if he/she keeps going to change lane the warnings take into account if the driver changes lane sufficiently in advance. One may look at Figure 2.14, to get a graphical sight on change lane maneuvers that generated warnings based on the time they are executed. Note that any warning is turned off as soon as you find the first strip in the open lane because the lane is free. In this case, the higher layer can provide a set of stop maneuvers to the action priming layer or that all maneuvers are safe (free lane).

Action priming

If the lane is open, this block just returns that all the initial jerks $J_{0,set} = \mathcal{R}$ are associated, and therefore, the warning is kept on level 0. If the ego vehicle is on a closed lane, the action priming block behaves analogously to the one of the VRU and WWD protection application 2.3.4. The set of safe maneuvers is the one to stop in the closed lane before the end (As we have seen, they also represent safe lane change in terms of starting points). Such set is represented by all the stop maneuvers with $s_f < s_{f,0}$, leading (thanks to monotonicity) to an initial jerk set as follows:

$$\begin{array}{rcl} M_{stop} & = & \mathcal{M}_{\text{CSS}}\left(a_0, v_0, s_f < s_{f,0}\right) \quad \Rightarrow \\ J_{0,set} & = & [-\inf, j_0(\mathcal{M}_{\text{CSS}}(a_0, v_0, s_f)) \end{array} \tag{2.8}$$

Examples of stop primitives sets are shown in Figure 1.4. We recall we use the stop maneuver to measure if the lane change is performed sufficiently in advance. For this reason, the mirroring approach is focused on the stop maneuver only.

2.4.4 Mirroring of intention

The mirroring is then focused on the safety distance from the endpoint of the closed lane, and the concept is: the lane change maneuver must be performed at a distance such that it is possible to stop before the endpoint safely. This metric allows for the maneuver to be made in time and to preserve the possibility to safely stop if the lane is occupied and it is not possible to perform the lane change maneuver. In Figure 2.14 some lane change trajectories are associated with the stop maneuvers used for mirroring. Three maneuvers are shown at the same velocity but different distances from the endpoint. One maneuver is instead performed with a higher initial velocity.

2.4.5 Experiment result example

while detailed and aggregated results are provided in 2.6.1. We now show an example of an experiment with non-equipped vehicles.
As shown in Figure 2.13, the layout is two parallel lanes, one closed and one open, each of them having 4 strips

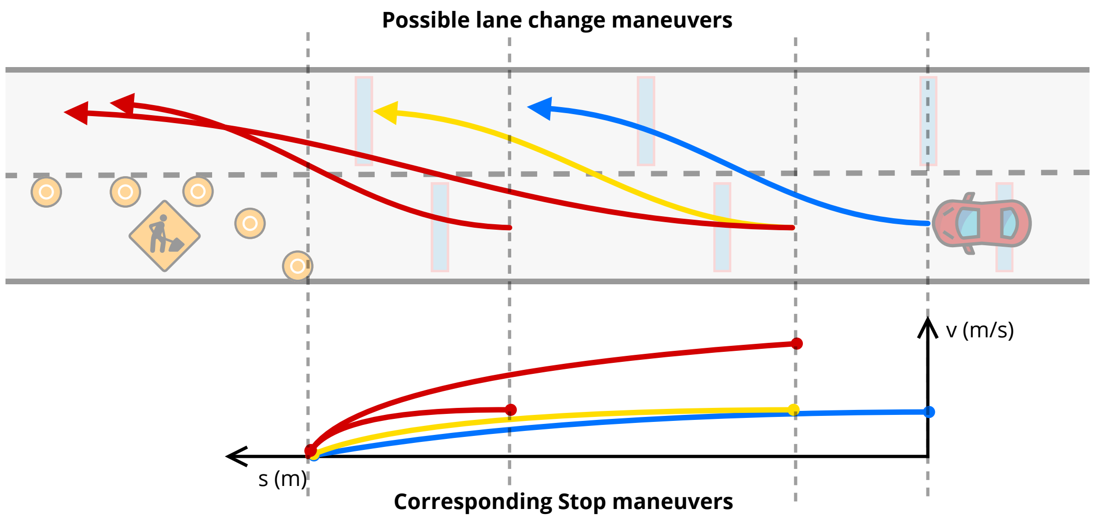


Figure 2.14: Example of Stop maneuver-based mirroring in the work zones case. The colors represent the warning levels (0,1, and 2), the trajectories are qualitative to show the concept. A maneuver is considered safe if performed before that stopping would become too dangerous. The trajectories do not represent the stop maneuvers; there are stop maneuvers on the plot if the vehicle could not change lanes.

up the final point of the closed lanes. Figure 2.15 shows an example of a road work scenario with two cars traveling in parallel lanes and one of the two is closed ahead. Figure 2.15 also shows the trajectories and the speed profiles obtained from GPS data only for performance evaluation purposes. In fact, for non-equipped vehicles. The warning is evaluated and issued only when the vehicle passes over the strip, indicated by blue vertical dashed lines in the speed plot. The warning is displayed on the HMI application installed on the mobile device with some delay, indicated by the black vertical dashed lines. The warning remains unchanged until the vehicle crosses the next strip in the lane for an update. The blue, yellow and red colors in the trajectory and speed profile plots correspond to the warning level. It is worth noticing that the vehicle in lane 2 (the open one) does not receive any warning. This is possible since the application is aware that the vehicle is in the right lane. On the contrary, the other vehicle keeps going at the same speed in the closed lane up to the third strip, which triggers yellow and then red warnings. If the vehicle in lane 1 had moved to lane 2 the Co-Driver would have computed the free flow maneuver instead of the stop maneuver (as for vehicle in lane 1), and no warning would appear. In other words, thanks to the lane level information provided by SAFE STRIP, the road works application evaluates the risk level consistent with the danger ahead and location of the vehicle proving the warning only to specific vehicles that need it. Further, some metrics are shown for some cases of non-equipped vehicles since it is pretty effective even if the vehicle state (e.g., position, velocity) is not provided continuously but obtained by the application through MQTT communication. Therefore, for the non-equipped vehicle, the Co-Driver generates the stops maneuvers only when the vehicle passes over the strips distributed before the closed lane.
In the case of equipped vehicles, the application works similarly, but it runs on-board and can take advantage of the continuously updated vehicle state. In this case, the Co-Driver runs at 50 ms, and the warning is updated at the same rate.

2.4.6 Experimental results

Even If the aggregated results are shown in Section 2.6.1, we decided to provide detailed results in this section since we need to differentiate over various cases. A reminder of the experiment structure is worth it to discuss the roadworks scenario results. The scenario is made of two lanes on a straight stretch of road where one of the lanes is closed by the presence of the roadworks, and the other is left open. The driver can decide to stop or to change the lane if possible. The roadworks experiments are divided into three use case sub-categories:

- A. Closed lane - one vehicle: one vehicle is running in the closed lane, and it can change lane instead of completely stop or slow down,

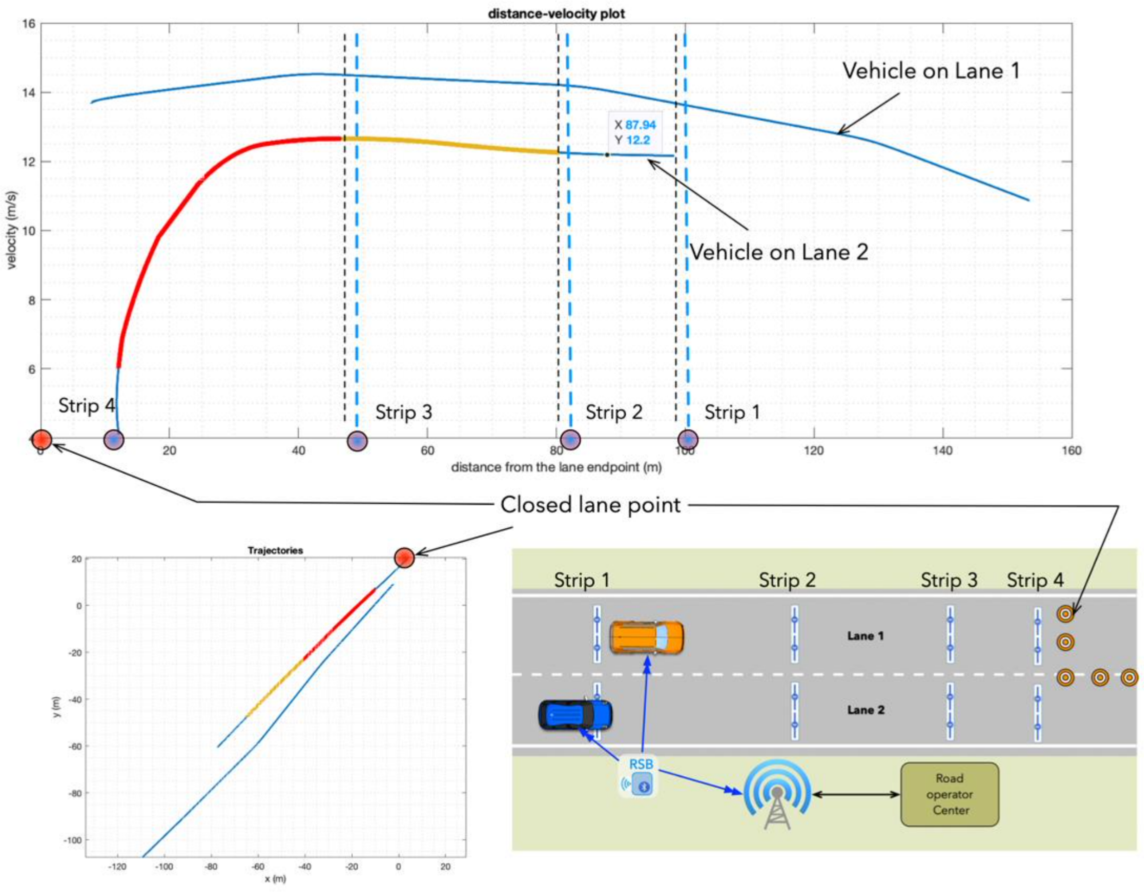


Figure 2.15: Example of experiment with two vehicles, one in the free lane, and the second is forced to stop. On the top of the figure the representation with velocity and distance, the color indicates the warning. In the bottom left the position in time, on the bottom right another schematic representation of the scenario. The vehicle on the free lane slows down for the warning level 0, while the vehicle in the closed lane is forced to stop since it cannot change lane. The latter vehicle reaches the red warning since it does not slow down (despite the yellow alert) till the final strip, when the red warning is triggered.

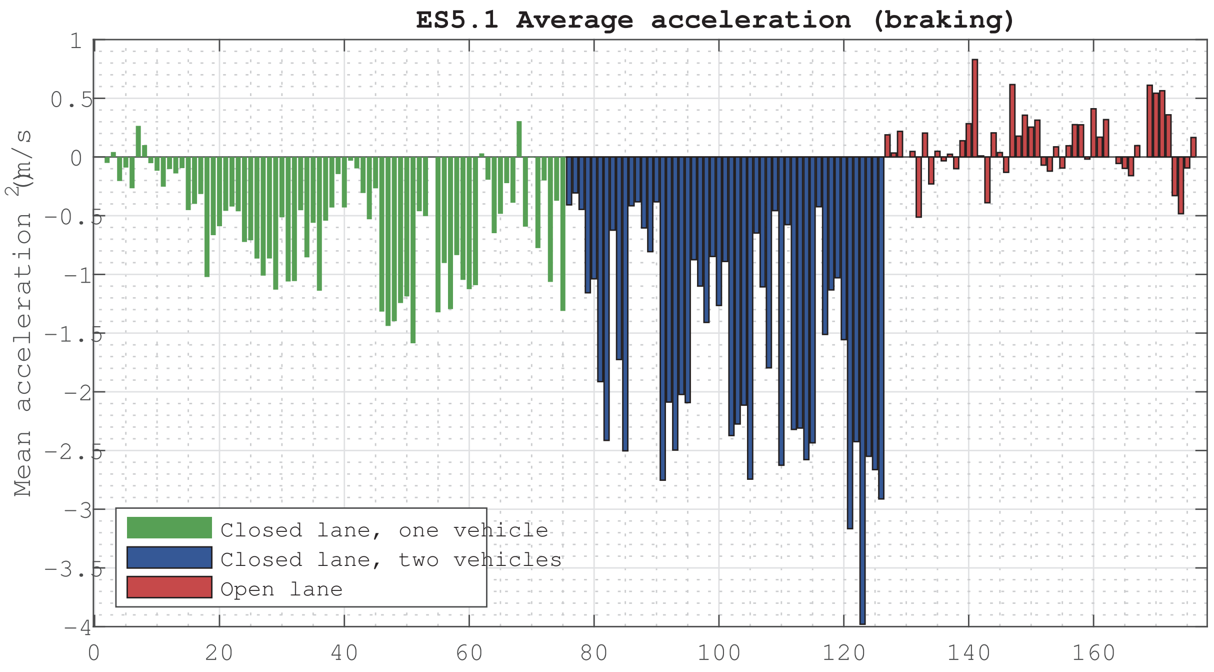


Figure 2.16: Average acceleration (negative), in the three different work zone use cases. In the first, the lane is closed but the vehicle is free to change lane since it is alone. In the second case the other lane is occupied by another vehicle, forcing the Ego-vehicle to stop. The last case is about vehicles in the open lane, that essentially do not slow down at all.

- B. Closed lane - two vehicles in the scenario, one is running in the closed lane and the other in the open lane. The vehicle in the closed lane cannot change since it is already engaged by another vehicle.
- C. Open lane: - one vehicle is already running in the open lane.

The average acceleration is particularly significant in this use case. While the maneuver time suffers from some variability, the average acceleration gives a better picture of the urgency of the maneuver. This is reflected in the variability among the different use above cases. For example, in Figure 2.16, this quantity is shown divided by the three cases.
In the SAFESTRIP project, the target value of deceleration for a smooth maneuver was 1.5 m/s^2. The case of the open lane is characterized by very low deceleration because essentially, the driver can keep going since he/she is only informed of the roadworks presence ahead (no check is done on actual speed even if, in principle, the Co-Driver can do it). In the case of a closed lane and one vehicle (blue bars), the expected threshold is fully respected: the driver had enough time to take a smooth corrective action after the warning. On the contrary, it is evident from the figure that in the case of an occupied lane, the average deceleration is most of the time above the target value. In order to investigate the compliance with the threshold, a one-samples and one-tailed t-test is performed between the values of these types of experiments and the target value required. The selected confidence is 5%. The test rejects the hypothesis that the difference between the average acceleration values appears to be not significant (p-value 0.35). The test results are summarized in Table 8.
Despite the test result, the orange bars in the plot suggest a significant amount of high values. To investigate further the data, this set of experiments is isolated and divided based on the driver's decision to stop or change the lane. The orange bars represent, in the sub-set of the two-vehicle, closed-lane case, the maneuvers that ended with the vehicle being stopped, the blue ones with a lane change manoeuver.
The plot shows that the stop maneuver requires a significant average deceleration, while the average deceleration is relatively minor if the driver decides to change lanes. A two-samples one-tail t-test confirms the difference. This test compares the mean values of the maneuvers with the "stop decision" versus the "lane change decision." The result is that the hypothesis is not rejected, and the mean value is different. The p-value is about 0.00026. In the tests in which the chosen option was to stop, the mean value of the acceleration is above the desired threshold by a significant amount. The null hypothesis is that the mean does not change. Different types of tests are performed to compare two sets of data and a set of data and a fixed value.

Test type	Data compared	H_0	p-value	Conclusions
t-test single-sample two- tails	Two-vehicles experiments average accelerations VS threshold (-1.5 m/s^2)	0	0.35	The overall mean of the two-vehicles experiment is not proven to be different from the threshold
t-test double-sample two- tails	Two-vehicles experiments average accelerations with change lane VS with stop	1	0.00025	There is a difference between the mean of the two cases “change lane” and “stop”
t-test single-sample two- tails	Two-vehicles experiments average accelerations with stop VS threshold (-1.5 ms-2)	1	$< 10^{-7}$	The mean value for the “stop” cases deviates indeed from the threshold

Table 2.2: Statistical test between different cases of the roadworks use case

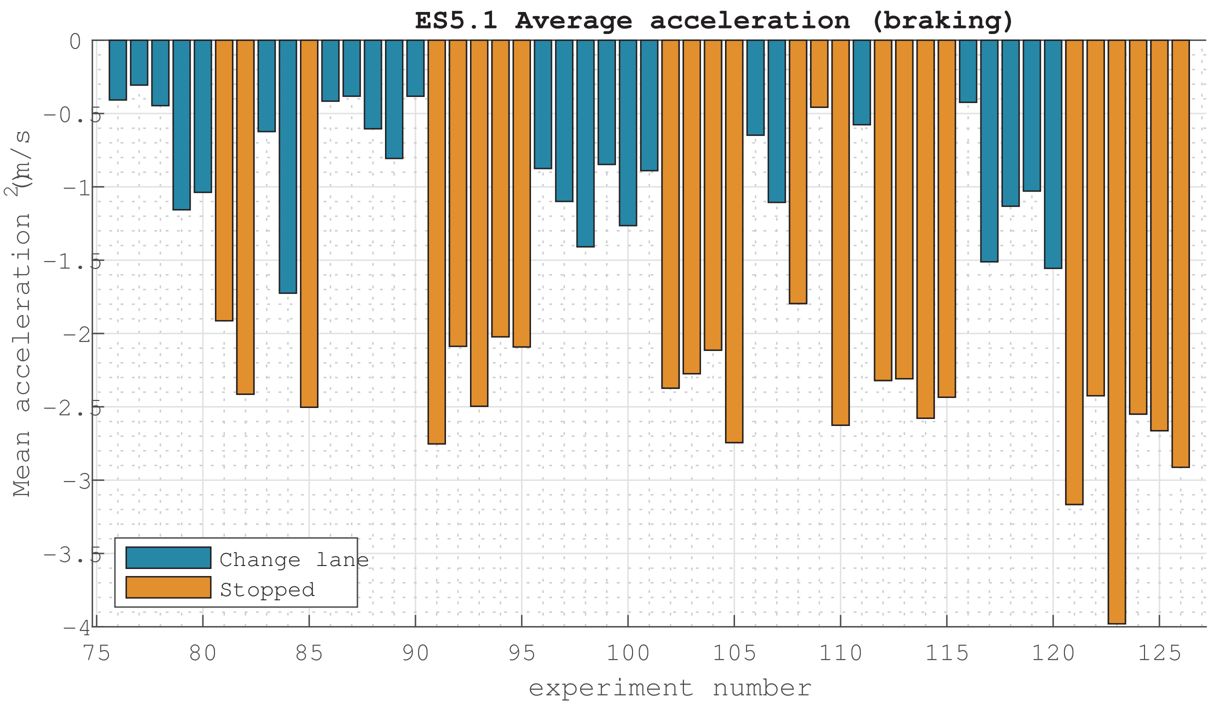


Figure 2.17: Average Acceleration indicator for vehicles that stopped in the closed lane and vehicles that changed lanes. Blue bars indicates those vehicle that changed lane.

2.5 Intersection support systems

We hereby describe the **intersection support system** based on the Co-Driver framework developed in SAFESTRIP project. This application addresses a challenging use case, even for state-of-the-art intelligent vehicles. The SAFE STRIP infrastructure is exploited to improve the performance of similar existing applications for connected vehicles, providing more accurate and rich information at lane level without the need for expensive on-board sensors. Most importantly, it enables the safety function on this use case for non-equipped vehicles. In this Section, after an overview of the existing solutions, we focus on the Co-Driver instance and the specific features of the intersection support systems, a more general explanation of the Co-Driver framework for road safety applications is provided in Section 2.2. Some simulation results and some specific experiments are discussed here, while the aggregated results for non-equipped vehicles are discussed in Section 2.6.1.

2.5.1 Overview of the existing solutions

The distraction of drivers is hazardous in the case of intersections and merging areas. A significant amount of accidents involves intersections and merging related traffic conditions, and many of them are due to an erroneous evaluation of the situation, and a consequent wrong action or behavior is executed. In several recent literature reviews on ADAS systems for intersections [102][29] most of the applications presented are based on optimization approaches where the safety has the higher priority, and other performance indices are hierarchically minimized or maximized like energy consumption, traffic throughput, comfort, and delays. The propose approaches are classified as centralized (i.e. [138]) or decentralized (i.e. [80]). A central system manages the vehicle traveling across the intersection to provide priorities and warnings in the first case. In the second case, each vehicle's intersection support system runs onboard, processing the information received from sensors and connected vehicles. Nevertheless, in both cases, they share the common requirement of data exchange using V2X. Data about the environment, road layout, and surface conditions can be collected by sensors onboard vehicles or by fixed sensors on the infrastructure. Centralized approaches tend to focus on monitoring the whole intersection, providing global optimum traffic management, and typically do not target the single driver. More driver-centered approaches (and decentralized) are based on machine learning models (reviews are reported in [42] and [10]). They are focused on understanding the driver's behavior through various data types, such as monitoring the driver using video cameras or inferring it from the vehicle's state.

2.5.2 Introduction

As stated in 1.2, the Co-Driver framework can perform the mirroring of intention (i.e., estimate what is the drivers' intended maneuver), and this idea has led to several applications, including the one presented in [34] and adopted in this project to be further specialized for the intersections to take advantage of SAFE STRIP technology. The objective of the application is to provide safety in general. This means reducing the number of accidents related to multiple vehicles, such as intersections and merging roads. More specifically, we want to address the problem of possible drivers' lack of attention or visibility concerning the presence of other vehicles. Therefore, the danger is related to other vehicles approaching the intersection, causing collisions. The scenario covered by this application is then *an intersection with other vehicles approaching*. Also, a merging ramp case was covered in the project, but we have not reported it here because the experiments were incomplete.
As for the other applications, the methodology is to warn the driver to make him/her aware of these dangers on the road.
The Co-Driver is expected to improve the system's acceptance, performing a similar intention estimation that humans would perform. The idea of the intersection and support systems, herein developed, is born as an extension of the work proposed on [34] the Co-Driver concept is introduced and used for intersection scenarios. In SAFESTRIP project, the application is extended with a more flexible warning logic, and the integration for the non-equipped vehicles. The latter is made possible by the SAFESTRIP infrastructure. For all the other applications, the Co-Driver evaluated the intentions of the driver of the ego vehicle. Here instead, the Co-Driver needs to evaluate also the intentions of other drivers. Thanks to the strips (or to the CAM message for equipped vehicles), the Co-Driver knows the state of the other vehicles. This information is employed to generate sets maneuvers for each actor in the intersection. Similarly to what it is done for the ego vehicle. After a more detailed description of the use case, the Co-Driver instance of this application is explained.

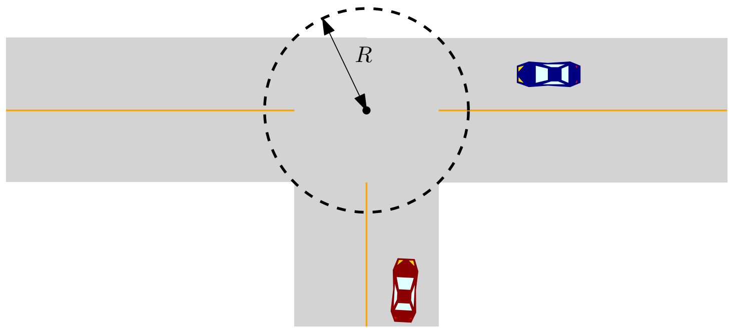


Figure 2.18: Example of intersection addresses by the application. The circle is the conflict zone, radius R can be tuned for safety margins. More complex geometries can be used in this application.

2.5.3 Intersection use case

As already stated, the intersection use case is the most complex scenario developed in the SAFE STRIP project. It involves multiple vehicles, and it is the only use case in which it is required to understand the intentions of non-ego vehicles. The use case in this project has tested with a maximum of two vehicles for experimental limits, but the application is designed for multiple vehicles, and tested in simulation.
The general layout of the use case is depicted in Figure 2.18. In general, we have a conflict zone that is indicated by the dashed circle of radius R, the collisions are supposed to happen only in the conflict zone and a sufficient condition to avoid collisions is that only one vehicle at a time can occupy the conflict zone. This conservative approach is justified by the nature of the application, which aim at providing warnings, the experiments validates this approach. The application is also designed to consider general topologies of the intersection under the condition that the vehicle can pass near a center point of the intersection. Therefore this excludes the roundabouts. Still, they could be treated only as separate intersections, one for each merging lane.

2.5.4 Intersection application design notes

This application is addressed in a slightly different way than in the others of Section 2.3 and 2.4. For the others, we always started from the data available from SAFESTRIP infrastructure. In this case we decided to start from a general application with mpre signals available. Then, we simplified the application in order to use it in SAFESTRIP. This difference is entirely encapsulated in the environmental elaboration block of the Co-Driver. Therefore, we introduce the general application, perform validation in simulation, and introduce the approximations for the SAFESTRIP version of the application. The experimental results presented in 2.6.1 are intended for the SAFESTRIP version of the application.

2.5.5 Co-Driver instance

The application structure, like the other applications, resembles the general one we have seen in 2.2 and is composed of two hierarchical layers (and three blocks). In this case, the outer layer interprets the intersection's geometry from the map message and tries to estimate the behavior of other actors (i.e., vehicles). This is done using the geometry of the intersection, the right of way priorities of vehicles crossing the intersection, and estimating the intentions of the other vehicles in terms of possible time to cross the intersection.
The inner layers are dedicated to inferring the ego-vehicle maneuver and assessing the risk level of the maneuver the driver is pursuing to provide a suitable warning. While the outer layer elaborates the environment and the intentions of the other vehicles.
Again we follow the Co-Driver mirroring approach: Interpret environment and other vehicles - generate safe maneuver - compare with the current state - send a warning.
The feasible maneuvers, which avoid collisions (i.e., are safe), are rated based on the effort necessary to execute them. It is assumed that the human driver will more likely follow the one that is easier to be executed. The agent estimates that the driver intends to select the minimum effort maneuver based on this assumption. And the task of the Co-Driver is to understand if this minimum effort maneuver is included in the safe maneuvers.

If the effort to pursue safe maneuvers is also proportional to the risk level, the higher this effort, the more the behavior of the driver deviates from safe maneuvers. In equipped vehicles, the process is continuously repeated to adapt to the changes performed by the driver, and a warning level (i.e., informative, advisory, and cautionary) is displayed to the driver when he/she does not react appropriately in non-equipped vehicles, the warning level is updated once any vehicle passes a strip. The maneuvers generated by this layer are compatible with the inferred intentions of the other vehicles and are considered safe in this respect. Those maneuvers are built using atomic longitudinal human-like maneuvers, the usual longitudinal motion primitives of Section 1.3. As for the other application, the initial jerk of the maneuvers is the direct metric for the effort required by the driver to start that maneuver.
Figure 2.19 shows the hierarchical structure of the Co-Driver instance. The environment elaboration layer elaborates the states of the non-ego vehicles and returns a pass goal with the residual time slots available for the ego vehicle. It also generates a stop goal, which is always a safe option. This block is described in Section 2.2.

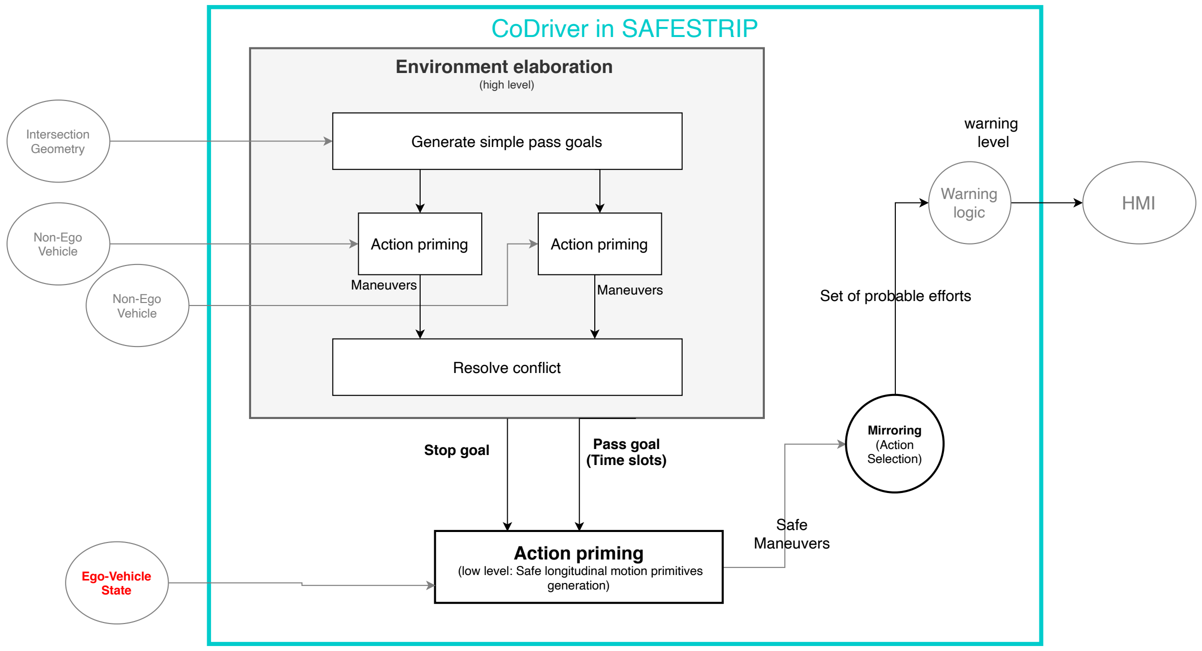


Figure 2.19: Layered architecture of the Co-Driver instance of the intersection use case. The environment elaboration contains other actions priming blocks associated to other vehicles in the intersection. The results of these blocks are collected and elaborated to carry out time slots fro the action priming block of the ego vehicle.

The hierarchical structure is shown in Figure 2.19. More details on the general Co-Driver architecture are reported in 2.2. Here we now discuss the specific roles of the blocks in the intersection application, with a particular emphasis on the outer layer that manages the conflicts between vehicles. One that is interested in conflict management as a separate application can refer to the work of the author in [117], which contains similar concepts of this specific Section. From now on, we will use an example layout of an intersection to illustrate the approach. It is depicted in Figure 2.18. The shape of a T-intersection is also the one selected for experiments.

2.5.6 Environment elaboration: conflict management

The environment elaboration layer plays a fundamental role in this application. Its task is to provide safe goals for the ego vehicle starting from the intersection geometry and the state of other vehicles. This block returns both of the goals seen in Section 2.2. They are **stop** before the conflict zone ($s_{f,0}$ is the distance), and **pass** within the time slots $\mathcal{T}$. The first goal is analogous to the other stop goals we have seen in the other applications (see Sections 2.3 and 2.4). The only difference is the distance to stop, which is computed as follows:

$$s_{f,0} = d_{r,v} - R, \tag{2.9}$$

where $s_{f,0}$ is the final distance for the stop goal, R is the radius of the conflict zone and $d_{r,v}$ is the plain distance between the vehicle and the center of the intersection.
To compute time slots instead, the same action priming procedure is applied for all the non-ego vehicles involved in the intersection to estimate time slots at which the non-ego vehicle arrives at the intersection. In the following subsections, more details on this steps are provided.
The time slots are time ranges when a vehicle can reach the conflict area. For Example, a time slot equal to the set [1,3] means that the vehicle can reach the conflict area between 1 second and 3 seconds after the current time. They can be the expression of permission to pass (the case of the time slots of the ego vehicle), or the estimation of the time the vehicle may arrive (the case of the time slots of the non-ego vehicles). The time the intersection is occupied is also considered, and it is explained later.
In this thesis, we also addressed the problem of autonomous vehicles intersection management in Chapter 4. The initial ideas for such a method were inspired indeed from the approach developed in this Section for the intersection merged with the optimization of the airplane landing in airports, (The reference for this topic is not available since it is still in progress).

The conflict zone

The first thing that is performed is the definition of an area of the intersection that is the one in which all the collisions may take place. The safety requirement is to free the area for a single vehicle to pass (i.e., it cannot be occupied by two-vehicle simultaneously). This area is called "conflict zone" or "conflict area" and we define it as a circle of radius R or R_cz, with the center in the geometrical center of the intersection (the position is provided by map message as reference latitude and reference longitude). This choice, which makes the solution conservative, is made because this project's scope is safety through the warning. The radius R_{cz} can be tuned according to data from the map message or tuned manually. The critical requirement of the conflict area must be the dimension, which has to be enough to guarantee safety if different vehicles do not share the area simultaneously.
In Figure 2.18 a qualitative example of the conflict zone is shown.
In the case of the autonomous vehicle instead (See Chapter 4), the conflict zone is tailored on the intersection with specific lengths defined for single conflicts, since the goal in such case is to improve efficiency (keeping safety obviously).

Time slots

A **Time Slot** T_i in this context is a range of time in which a vehicle i can **enter** the Conflict Zone (CZ). This approach is all about estimating and cutting time slots. Thus, the mirroring of other vehicles' intentions consists of estimating their "unconstrained safe maneuver," leading to estimating other vehicles' time slots. The estimation of the ego vehicles' time slots (necessary for the action priming block) is performed considering the following factors: all the possible feasible trajectories of the incoming vehicles, compelling conflicts between trajectories, and the right of way order in the intersection. Finally, the computation applies an action priming block seen in the previous Section on the other vehicles given the initial conditions and takes all the feasible maneuvers. At this step, some assumptions are made:

- A maximum velocity v_{max} at which the intersection can be crossed is selected; it does not have to be necessarily the speed limit of the road.
- The trajectories associated with the same vehicle can be considered coincident out of the conflict zone R (see Figure 2.18).

For each vehicle approaching the intersection, all the feasible trajectories are instantiated; since the geometry of the intersection is fixed, the feasible trajectories do not change over time during the crossing. The reference trajectory is generated as a sequence of clothoid preserving continuity in curvature; this aspect will not be detailed in this Section for space reasons. We assume that vehicles do not significantly deviate from these trajectories, but this is not restrictive if we consider sufficient safety margins. Once the trajectories are defined, the action priming block needs the available time slots, according to the other vehicles, the right of way order, and the conflict between trajectories.

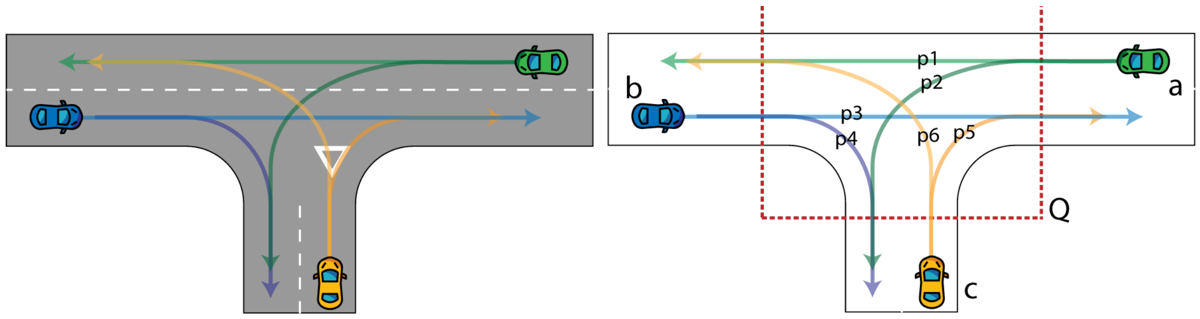


Figure 2.20: T-Intersection with six paths and three vehicles. The symbol p_k indicates the path k. Each vehicle can be associated with several paths, since we do not know in advantage which paths is going to follow.

Safe condition

In order to avoid a collision, the constraint we impose is that no vehicle can share the conflict zone with another vehicle simultaneously. And this condition will be used for the computation of the time slots.

Computation of time slots

The computation of the time slots is performed in several steps. This calculation requires the longitudinal states (acceleration, velocity, and position w.r.t the intersection) of the vehicles involved. A scheme of the algorithm is shown at the end of the Section. The first step is to instantiate one vehicle for each trajectory (from p_1 to p_6 in Figure 2.20); those vehicles (called virtual vehicles) are needed because the trajectory choice of each vehicle is unknown; for Example, the vehicle a in Figure 2.20 can choose to follow the path p_1 or p_2. Therefore, for instance, vehicle a is associated with the two virtual vehicles corresponding to path p_1 and p_2. For those virtual vehicles, we define the points of intersection between the conflict zone and the trajectory p_i, and they are the entry points and the exit points of the conflict zone. Concerning the entry point, we define a curvilinear abscissa $s(t)$ on the trajectory p_i as the longitudinal position of the virtual vehicle, with velocity v and acceleration a. They compose the longitudinal state of the virtual vehicle, just as we have already seen for all the other applications. This state coincides with the one of the actual vehicle associated. The following steps are performed to resolve vehicle conflicts and inhibit overlapping time slots. First, the possible intersections of all the trajectories are manually detected (only once, before deploying the system). No conflict is considered between trajectories that share the same starting lane (in our use case, the couples 1-2,3-4,5-6). After this step, the right of way order is taken into account to establish the order of passing for each conflict. Our use case was chosen to use the "priority to the right" rule and the yield road signal for the vehicle. To represent the conflicts, the right of way matrix is defined (M_{rw}). The concept of the right of way matrix is often used in traffic simulators (e.g., SUMO [43]), (M_{rw}) is a square matrix, where the columns and the rows represent the possible paths. The entry $M_{rw}(i, j)$ of the matrix represent the conflict state between the trajectory p_j and the trajectory p_i, the entry $M_{rw}(i, j)$ is equal to one if there is a conflict between paths p_i and p_j and path p_i has the right of way. If it's p_j that has the right of way, the value is -1. When there is no conflict, the entry is equal to 0. The matrix is anti-symmetric, since for each conflict only one trajectory can have the right of way on the other. Note that the right of way matrix can be computed using fixed rules from the geometry,roadmarks and signals. For our use case example (Figure 2.20) the following M_{rw} matrix is defined:

$$\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 1 & 0 & -1 \\ 0 & -1 & 0 & 0 & -1 & -1 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{Bmatrix} \tag{2.10}$$

For instance, the path p_2 intersects paths p_3,p_4 and p_6. In the first two cases, the path has no priority. In the last case, the trajectory has priority. In our application, the right of way order is meant to be provided by the infrastructure.

Vehicles and order to pass

The right of way matrix now is used to inhibit the time slots of each vehicle, cutting them with the ones of possible vehicles with higher priority. In order to facilitate this operation, the inhibition of the time slots must be performed in a compatible order of passage In this way for each trajectory the time slots with higher priority must be available for the computation of the next one. This condition can be formalized as follows:
Given σ a vector which entries are an order that is compatible with the priority of the paths. Given $\boldsymbol{i}^*(j)$ the vector of indexes of the entries with value -1 in the vector composed by the column j. The following condition must hold:

$$\sigma_{\boldsymbol{i}^*(j)} < \sigma_j \quad \forall i, j | M_{rw,i,j} = 1. \tag{2.11}$$

The condition above is equivalent to reordering the column of the matrix M_{rw} in the order σ in order to collect all the -1 in the upper triangular.

If the time slots computation is executed according to the order σ, time slots of higher priority vehicles are available for cutting time slots of lower priority vehicles. In our example the order is $\sigma = [1, 3, 4, 2, 5, 6]$. Different vectors σ are possible if there are no pairs of paths with no conflicts. Moreover, those vectors do not necessarily represent the passing of pass. The vehicles that have to yield to others can pass before respecting the time slots, and this happens if the priority vehicles are far enough. The next step is to define the maximum and minimum velocity to generate feasible initial time slots; this is performed using the same algorithms in [1] starting from one wide time slot (0-1000s). Here the trajectories data are used to discriminate the v_{max} value: the maximum velocity is computed according to human tolerance criteria called two-third law shown in [24] which depends on the curvature (ρ). It is chosen (according to [24]) to represent the 99.9 percentile of the human drivers (according to [24]). The maximum velocity is limited on a feasible one (can be higher than the legal limit of the road).

$$v_{max} = \min\left(v_{max}, v_\rho\right) \tag{2.12}$$

Time slots can be initialized and then cut for all the virtual vehicles in order σ. The first initialization considers only the provided bounds on velocity, like the intersection was free. The inhibition consists of cutting the time slots of the higher priority virtual vehicles. The time inhibited in this phase is only the set of the possible arriving time of the higher priority vehicles, representing all the instants they can arrive at the intersection.

Crossing the intersection

Until this point, we only considered the time to arrive for the vehicles. However, the time to cross the conflict zone is nonzero. We assume a constant velocity in the conflict zone (this velocity is updated at every step to avoid collisions due to the slowing down of vehicles, but this aspect was not validated). The distance in the conflict zone is computed for each path p_i. We define the times to cross as follows:

$$T_k^{cross} = v_{k,max}\delta_{s,k} \tag{2.13}$$

Where $\delta_{s,k}$ is of the portion of the path p_k inside the conflict zone. This time to pass the intersection must be then taken into account in the inhibition (T^{cross} in Figure 2.21. For the ego vehicle to pass, the first time slots to inhibit is that the ego vehicle itself needs to clear the intersection.

Final time slots

The time to cross must be concatenated before any time slot of the other vehicles to ensure the intersection is clear if the ego vehicle passes before others with higher priority. The time to cross the other vehicles must be instead concatenated immediately after the last arriving time of them because the ego vehicle has to wait for the other vehicles to clear the intersection to pass. The time to cross is estimated using the current velocity of the vehicle over the residual arc of the trajectory in the conflict area. This choice was made to be robust in case of an unexpected sudden slowdown in the vehicle in the conflict area.

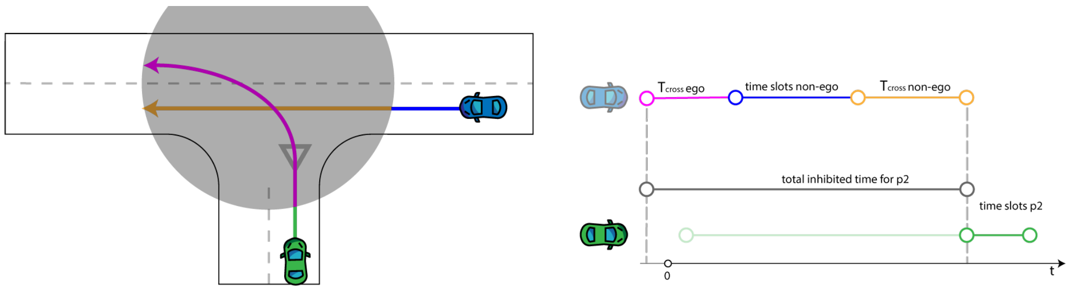


Figure 2.21: Example of time slots inhibition for ego vehicle (green). The available time slot for the green vehicle is cut by three quantities (for one opponent vehicle): the time to cross of the ego vehicle, if s/he wanted to passes first, the time slots of the opponent vehicle, and the time to cross of the opponent vehicle, since the ego-vehicle must wait the conflict zone to be free. The concatenation of this quantity always starts from the time slots of the opponent vehicle in this case because of the priority. We provide more details in Algorithm 2, where one can replace i as a generic ego vehicle and j as opponent in order to understand the procedure.

The time slots computed, are used for the computation of the maneuvers and the warnings. The algorithm allows to compute time slots for each virtual vehicle involved and for the action priming block the time slots are the one of the ego vehicle. The algorithm is summarized here:

Algorithm 2 Computation of time slots

```
Input: Vehicles states $s_j$ $v_j$ $a_j$
 1: Instantiate virtual vehicles and trajectories and assign states
 2: Compute trajectories conflicts and generate/write $M_c$ Matrix
 3: while true do
 4:     for $i \in \sigma$ do
 5:         Initialize time slots $T_i$ using $v_{min}$ and $v_{max}$
 6:         compute $T_i^{cross}$
 7:         for $j \in j | M_{rw,i,j} = 1$ do
 8:             get $T_j^{cross}$
 9:             Add $T_j^{cross}$ after $T_j$
10:             Add $T_i^{cross}$ before $T_j$
11:             Cut the resulted slot from $T_j$
12:         end for
13:     end for
Output: final time slots $\mathcal{T}_j = T_j$
14: end while
```

In the end, this block generates two goals: the goal to pass in the time slots and the goal to stop before the intersection. The latter is always a safe option since collisions are eventually avoided stopping before the intersection.

Action priming

As in the other applications, the Co-Driver generates safe longitudinal maneuvers that the driver can pursue at this stage.
As stated above, we have two goals to fulfill with maneuvers. The first is the stop goal. The point to stop is the initial point of the conflict area, and its distance is again defined as $s_{f,0}$. Using the notation in Section 1.3, we generate the "usual" set of stop maneuvers, yielding to the set:

$$M_{stop} = \mathcal{M}_{\text{CSS}}(a_0, v_0, s_f < s_{f,0}). \tag{2.14}$$

The other goal is the *pass* goal. The other set of maneuvers then aims to reach the intersection (the conflict zone) at a given time T, which has to be in the time slots $\mathcal{T}$. The final velocity is not constrained, and the final acceleration is zero. The final velocity depends directly on the maneuver's duration and constraints space. The final velocity is lower bounded to (3 m/s) because too slow maneuvers are too similar to stop maneuvers (The same velocity bounds are used in [34]). For the pass goal, we use the primitive called CST in Section 1.3.

$$M_{pass} = \mathcal{M}_{\mathrm{CST}}(a_0, v_0, s_{f,0}, \mathcal{T}). \tag{2.15}$$

maneuvers can be instead represented by all the primitives with final time in the time slots, condition expressed by $T \in \mathcal{T}$, where $\mathcal{T}$ is the set of the time slots and it is a union of compact subsets. For example, $\mathcal{T} = [T_1, T_2] \cup [T_3, T_4]$ represents the possibility to pass in one of the two intervals. To summarize, the complete set of the maneuvers is the union of the two sets, indicated by M_{int}.

$$M_{int} = M_{stop} \cup M_{pass} \tag{2.16}$$

It is important to remark that a set of maneuvers (for our purposes) can be defined knowing only the maneuver at the bounds of the set. Therefore, we do not need the expression for all the possible maneuvers (as stated in Section 1.3.6). In this application, we are interested in the initial jerk. Thanks to the monotonicity of the initial jerk with time (and space for the stop maneuvers), jerk sets are defined by their bounds maneuvers. An example of maneuver generation (actually completed with mirroring of intention) is depicted in Figure 2.22.

Mirroring of intention (Action selection)

As we have already explored twice, we must now to generate a proper warning level for the driver when s/he is not acting safely. Minimizing the effort is preferable and closer to the driver's choice among all possible ways to execute the maneuvers. The two sets of safe maneuver described in (2.16) generates a set of initial jerks. thanks to the monotonicity of $j(0)(T)$ and $j(0)(s_f)$ in the stop and pass maneuvers, the sets in (2.16) correspond to one unbounded set for the jerk and to a union of compacts for the other ($J_0(M_{pass})$):

$$J_{0,set} = [-\inf, J_0(M_{stop}(a_0, v_0, s_{f,0}))] \cup J_0(M_{pass}) \tag{2.17}$$

At this point, the mirroring is identical to the other approaches and to the general Co-Driver architecture we have seen in Section 2.2. If the safe jerks (the set bounded by the thresholds for each warning level) overlaps, the corresponding warning level is not issued. An example is provided in Figure 2.22.

2.5.7 Simulation validation

As stated earlier, we developed a general approach that can deal with several intersections, imposing a right-of-way order and handling the time slots hierarchically. We implement simulations in which the vehicles follow the time slots to cross the intersection to validate such logic. However, this is not possible (and out of scope) in the SAFESTRIP context. Therefore, we provide a simulation framework and result to test the presented time slot logic.

Simulation environment

The simulation environment was developed in Matlab (2020b). A single-track vehicle model was used to simulate the vehicle dynamics. The algorithm is executed periodically every 100 ms. Two different cases are considered for the scenario in Figure 2.20. The difference is in the driver's behavior.

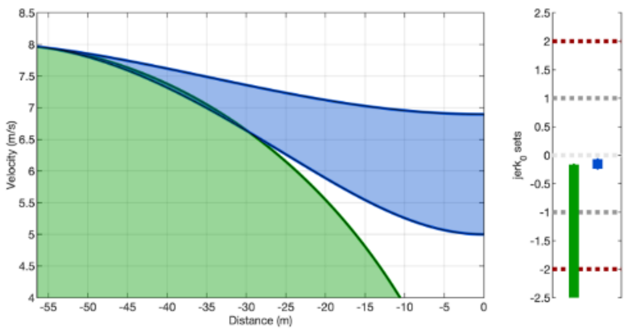


Figure 2.22: On the left, a set of stop maneuvers and a set of pass maneuvers from the action priming block (from actual values). The plot on the right only represents a vertical axis with the jerk in ms^{-3}. It shows in blue and green the two subset of the final J_0, the colors indicates the maneuvers they come from. On the same plot, dashed lines are the bounds of the warnings' jerks sets for the mirroring process. In the selected case, the warning level is 0. A detailed explanation of the warning strategy is in Section 2.2, as well as an example in Figure 2.5.

1. Full autonomous case: in this case, the vehicles are driven by the Co-Driver, using maneuver one selected the first available in the (pass) safe maneuver set. This case aims to validate the algorithm's effectiveness in terms of maneuvers generation and time slots calculation. In this case, the maneuver selected is the slowest one of the first (pass) time slots available.

2. Warning-based case: Vehicles a and b are considered autonomous (or modeled by the Co-Driver), and vehicle c is represented by a decision-making model of the driver. The model consists of two case switches. Unaware behavior: the driver is unaware of the other incoming vehicles, and the initial velocity remains the same until the behavior switches to "aware." Aware behavior: when the Co-Driver issues warning, the reaction time of the driver is taken into account (value based on [34]), waiting 1.5 seconds before switching on the Co-Driver safe maneuver (which can be a stopping maneuver or a pass. This case is aimed to evaluate the warning anticipation; the vehicle "c" was chosen because it is the one with most of the conflicts in which it has to yield to other vehicles.

Simulation results

For case 1, 50 simulations were performed with random initial conditions and trajectory choice. Distance and velocity from the intersection (the conflict area) were generated in the range 50-100 m and 8-16 m/s. Any vehicle chooses randomly one of the two associated paths (for instance, vehicle 1 can choose between p1 and p2). No collisions occurred between vehicles in the simulations. This result is shown in Figure 2.21. Every chart is associated with two conflict paths (6 plots), in the y-axis, the number of the experiment is indicated, and for each of them, the time at which the intersection is occupied is shown. Only the case when the associated paths were selected is shown in each plot. The absence of contemporary occupancy of the conflict area is shown in Figure 2.21, between the crossing times is always present a safe gap, which is not explicitly imposed in the algorithm, but it raises from the fact that the time slots are considered for all the possible approaches velocities, to be robust to the practical maneuver choice. In the majority of the cases, the maneuver with the right of way (the blue one) passes before the other. The opposite situation happens only with a significant time gap because the yielding vehicle considers its own time to cross.
For case 2, the 50 simulations have the same random initial conditions, except for the distance which is generated in the range 100-200 m. Figure 8 is shown the time to an intersection (TTI) at the instant the warning is issued. TTI is defined as the time to reach the bound of the conflict zone keeping the same velocity $TTI(t) = v(t)/s(t)$. In Figure 2.24, blue triangles represent the case in which the driver is the last passing the intersection, the green represents cases in which he/she passes before at least another vehicle. In most cases, the TTI is linear with the velocity, which is the case of the theoretical warning time in [1]. Cases in which the driver can pass before occur when the warning is generated, but there are available time slots to pass before. This can happen only if the driver has high velocity and the others are slow. There are 2 cases of $TTI < 2.5s$, those are cases in which the others rapidly clear the intersection, but the velocity is too high for the curvature of the trajectory, a consequence of the implementation of the two-third law [24] for the accepted velocity.

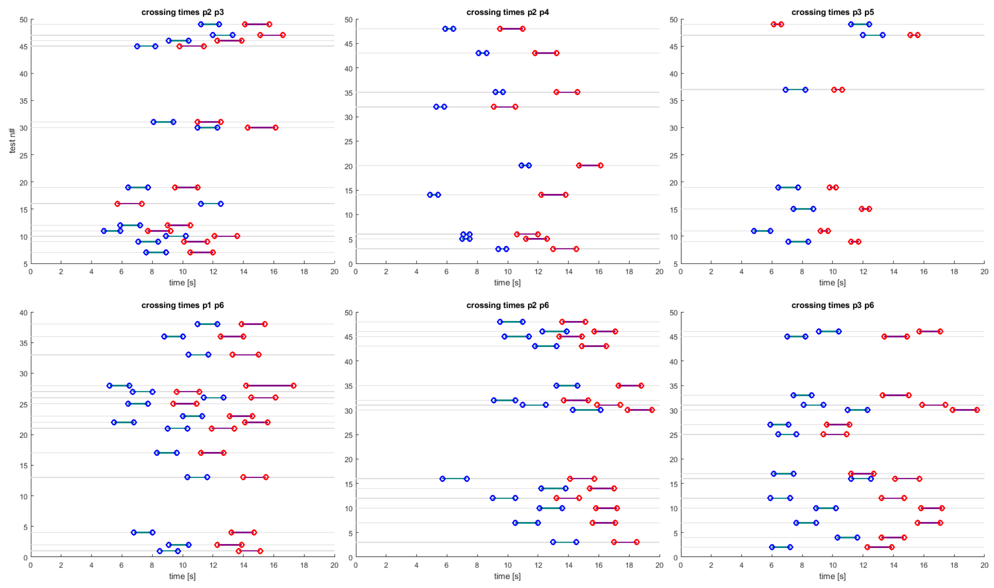


Figure 2.23: Crossing times for each case of conflict, in blue the time frames of vehicles with right of way on the others, the segments represent the times frame the correponding vehicles passes in the conflict zone.

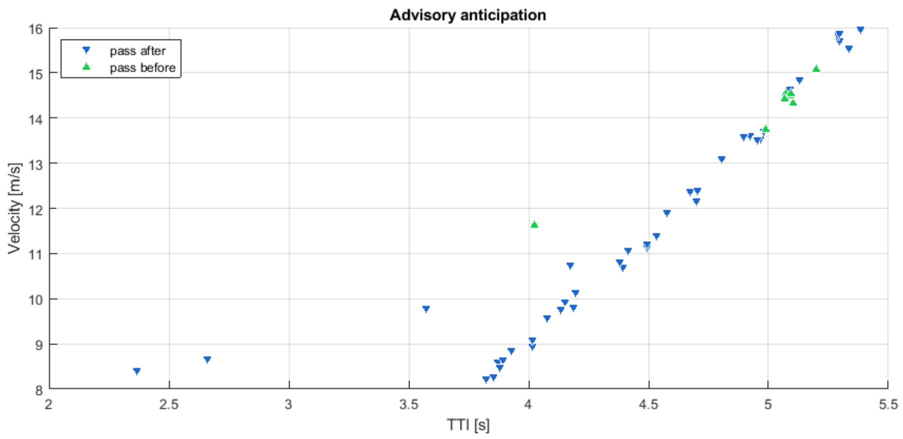


Figure 2.24: Time To Intersection for warnings of “non-aware” driver

2.5.8 SAFESTRIP specific features and approximations

The application illustrated and validated in simulation is now adapted to the two use cases that we have seen (intersection and merging). However, to do so, we need to make some further assumptions and introduce a feature for the reaction time compensation.

Time slots in the SAFESTRIP case

In the SAFESTRIP case, we decided to consider all the opponent vehicles intended to pass. This is because the application is focused on safety, and we are interested in the effectiveness of the warning in case of danger. It is also worth noticing that if the conflict zone is not too conservative, the mirroring of intention should also model an opponent vehicle that slows down to stop. This action may leave a wide enough time slot for the ego vehicle to pass, even if the opponent has the right to pass. The other approximation regards the trajectories, we consider the trajectories as rectilinear, and the segment inside the conflict zone is taken conservative as the diameter of the conflict zone. (For time to cross computation mainly.) Apart from these two aspects, the application does not change from what we have explained in previous sections.

2.5.9 Experimental validation

This Section now addressed the experimental results of the application. As for the other applications, we start with a minor experiment set using "equipped vehicles, provided with V2X and a precise positioning system. This is to validate the application in an ideal case. After that, we move on to a more extensive experiment set on the non-equipped vehicles. That validates the application in the scenario where the only information comes from the STRIPS on the road, and the communication is performed using a smartphone and the MQTT framework.

2.5.10 Equipped vehicles results example

For the equipped vehicles, The plot in Figure 2.25 shows the drawing of the actual layout and the trajectories of the vehicles colored with the warning levels that were issued in the experiments. Different colors correspond to different levels of warnings. For Example, it is possible at first glance to see that the red is closer to the center of the intersection than the yellow, which means it was issued just before the intersection for a critical case after a progression from green to yellow. In Figure 2.25 the trajectories obtained in the equipped experiments are shown. The colors on the trajectories indicate the warning level. This plot is possible only for the equipped vehicles experiments since the position was provided with enough precision and rate. The variability of colors among trajectories is due to the vehicles' actual velocity and acceleration value being the Co-Driver designed to take them into account. In the intersection case, the warning also depends on the opponent vehicle. For Example, some curves are completely blue since the opponent vehicle had cleared the intersection before the go vehicle was close to it. Some blue colors after the intersection are due to the vehicle heading back to the intersection and triggering the system in new intersection management.
Experimental results are detailed in 2.6.1, aggregated with the ones of the other applications.

2.6 Experimental results

In this section, we collect all the results in terms of Time To Collision and Average Acceleration of the experiments. Results on the telecommunication performance and the electronics are not reported here, since they are out of the scope of this thesis.

Metrics

The metrics used in this section are the one used in the project to evaluate the timing of the warnings and the driver behaviours (average acceleration after warning).

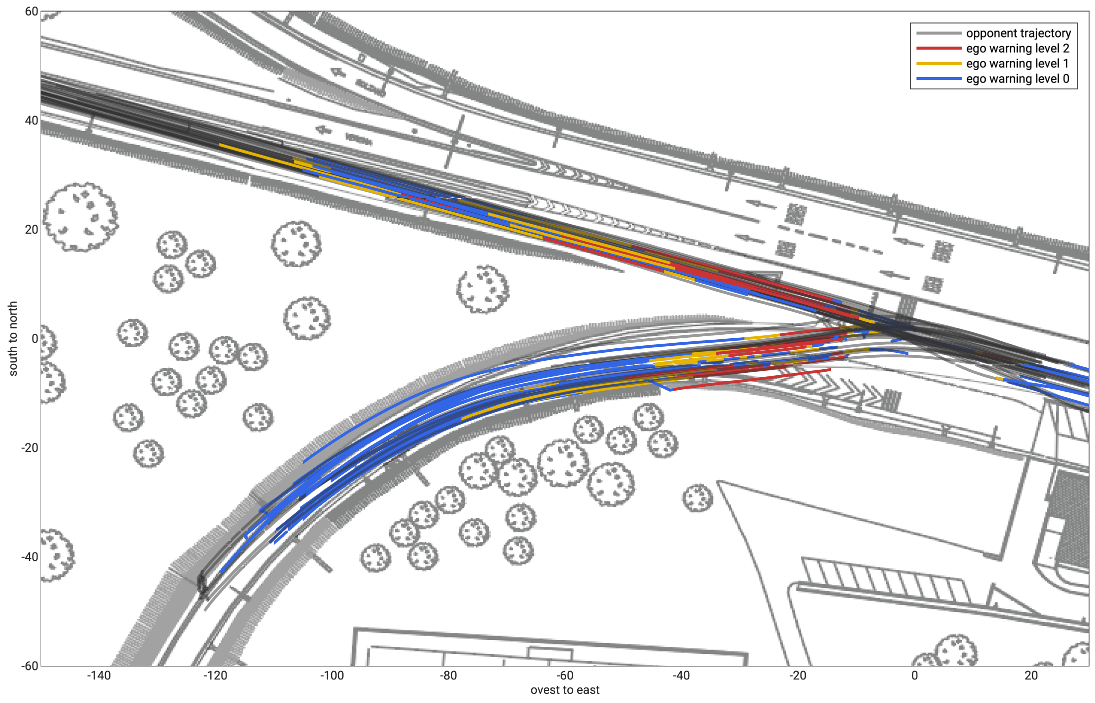


Figure 2.25: Experimental trajectories with warnings in different colours, blue means level 0, yellow level 1 and the higher value is represented by the colour red.

Time To Collision (TTC) It is computed as:

$$TTC = \frac{v(t_0)}{s(t_0)} \tag{2.18}$$

Where t_0 is the time the first warning was issued and v and s are velocity and distance respectively.

Average acceleration (a_a) is the average acceleration during the maneuver time, it is computed as:

$$a_a = \frac{v(t_0) - v(t_1)}{t_0 - t_1} \tag{2.19}$$

where t_1 is the time the danger is not incoming anymore (warning level deactivated, not even 0)

The results are divided by performance index (Time To Collision and Average Acceleration) and by kind of vehicle, equipped and not-equipped. Finally, the result is divided by use case. We hereby formalize some acronyms to indicate the use case:

VRU This acronym means Vulnerable Road User. This use case is discussed in Section 2.3, and consists in pedestrian protection at the zebra crossing.

WWD Wrong Way Driving as the name suggests, it concerns the protection from a vehicle arriving in the wrong way from a ramp. The wrong-way vehicle is actually detected at the ramp's beginning (end).

RDW The roadwork use case is the only one tested with a len personalization. It concerns safe behavior in the presence of one or more lanes closed for roadworks. Note that in these results, all the cases in which the vehicles started in a free lane are not included since the system did not provide any warning.

INT It is the intersection use case. It is the most complex one, and it does take into account the behavior of the other vehicles in a detailed and dynamic fashion. In this case, warnings do not depend on static obstacles.

In Figure 2.26 we can find the boxplot of the metrics. Discussions are provided in the next sections.
Additional experiments were made to cover two aspects: the Powered Two-Wheeled vehicles, and on-board GPS

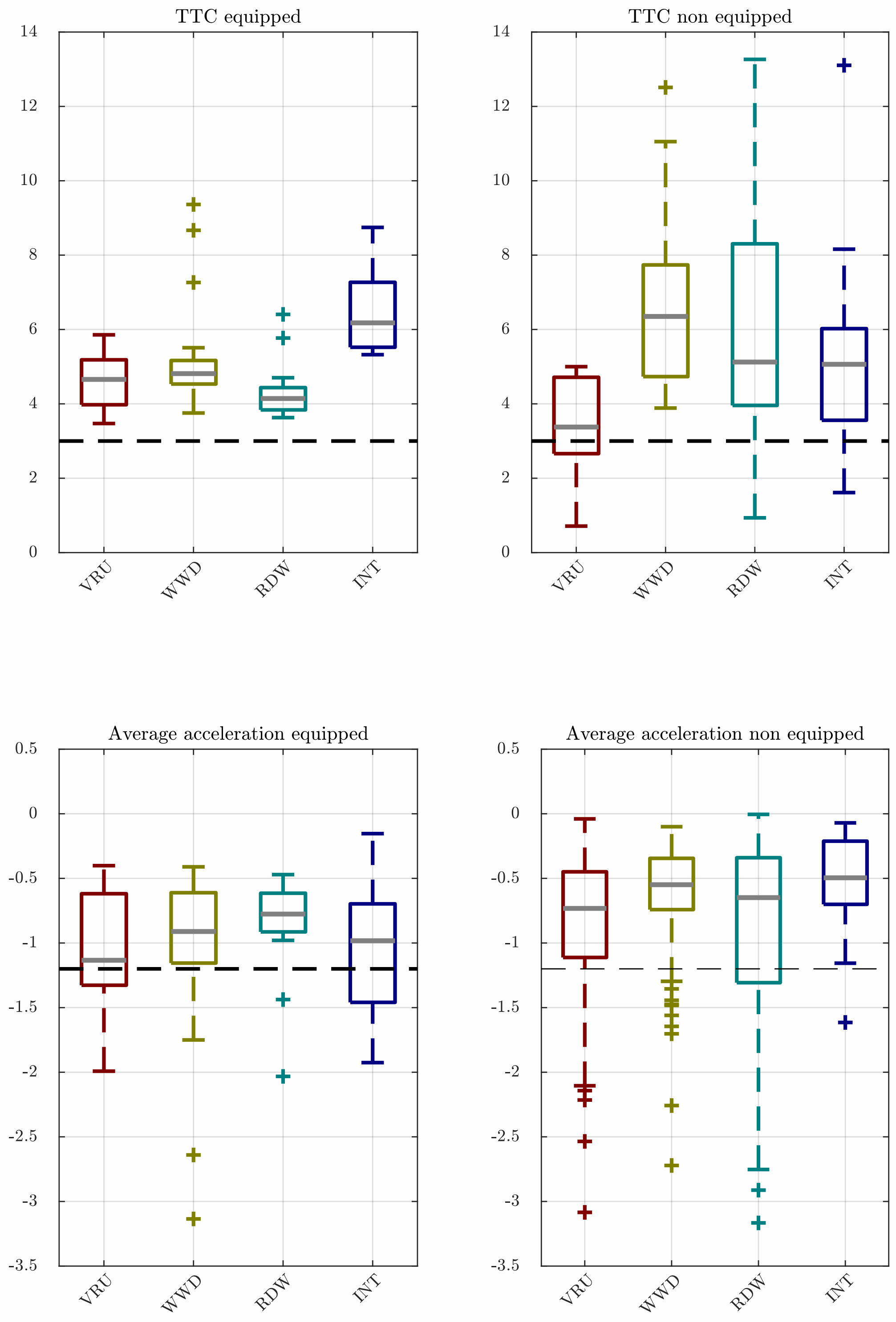


Figure 2.26: Aggregated results: BoxPlot of the metrics for each use case, divided by equipped and not equipped

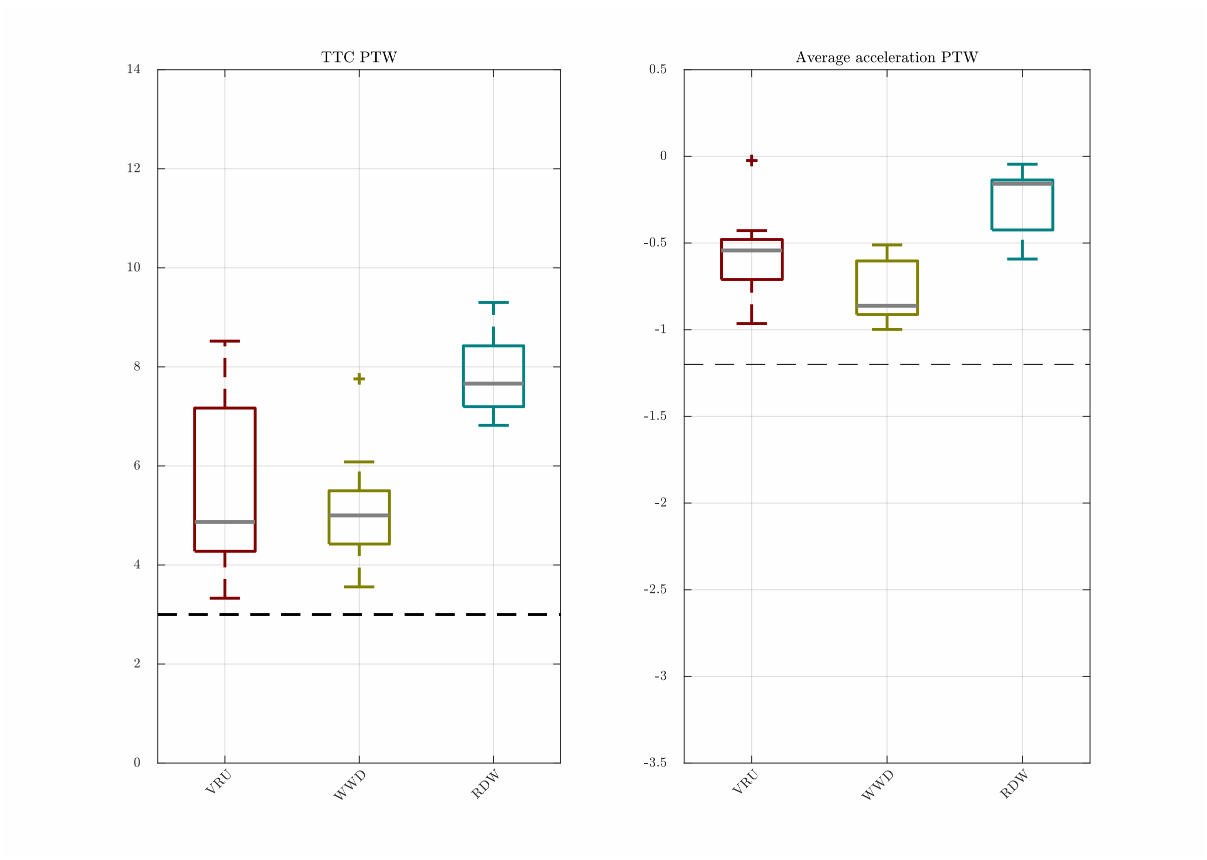


Figure 2.27: Aggregated results for Power Two-Wheeled vehicles. Again divided by use case.

and IMU sensors communicationg with LTE. Main purposes of these experiment were to validate the usage of the MQTT via LTE communication paradigm (Not affected by the asyncronous communication from the strips). And to test the system with PTWs. Availability limits do not allow us to test with PTW in other than this way. Summarizing, PTW experiments provides the state of the vehicle (as well as the STRIPS) with on-board sensors through the LTE connection. Results are provided in figure 2.27 and discussed in the next Section.

2.6.1 Results discussion

Time to collision of the equipped vehicles

In Figure 2.26, the TTC are shown int he form of box plots. To meet the requirements of the SAFESTRIP project [5] we expect a time to collision behind 3 seconds. This reflects the reliability of the communication for the equipped vehciles, the V2X standard is indeed designed for low-delay communication, accomplishing safety critical tasks. The few outliers correspond to a more conservative TTC. The TTC indicator does not include the acceleration in the computation, while the Co-Driver does. The Co-Driver issues the warnings as soon as is able to detect the intentions according to the algorithm seen in 2.2. The Co-Driver takes into account that the initial high acceleration increases the effort of the driver to pursue a safe maneuver slowing down, thus, those outliers correspond to particularly critical behavior that required an earlier warning. Let us now look at the intersection use case. The TTC is higher in general and we can recognize a higher variability. The higher mean (or median) is due due to the whole area that is accounted as a source of danger, while the collision for the TTC is the center of the intersection. This means that the actual distance for the maneuver of the Co-Driver is shorter than the distance to the center of the intersection, leading to anticipated warnings. The variability of the TTC concern the behavior of other vehicles. Changes of intention of other vehicles (there was only one other vehicle in the experiments) lead to change in the set of safe maneuvers, for instance if the opponent vehicle accelerate (even slightly) towards the intersection. The time slots we have seen in Section 2.5 are dynamically updated and therefore the warnings may happen in different time frames w.r.t. static obstacles.In general, the TTC is used to evaluate to the performance of the system: we expect it to be similar or larger than the expected value of 3 seconds, shorter values would indicate warnings that started too late, even if they made sense in terms of driver intention, but the idea was to change the intention of the driver in time.

Average acceleration of the Equipped Vehicles

The average acceleration metric (in absolute value) is more about the behavior of the driver. It helps to understand how the driver reacts to the warnings and how sudden he/she changes intention to a safe one. If the driver is warned ineffectively or too late, this metric will exceed the proposed threshold (1.2 m/s). The average acceleration exceed the threshold for a significant number of case, the median and the mean do not exceed it instead. The outliers represent some cases of over-reactions for WWD and RDW applications, while in intersection and VRU applications we have a higher amount of higher deceleration (in absolute value). We attribute this behavior to the fact that zebra crossing and intersections are everyday situations and the drivers may be more confident in decelerate later. The system is considered particularly effective in these scenarios where the source of danger is not visible and in more extraordinary situations. Especially the WWD represents (fortunately) a very rare event in regular driving, the systems seems to allow the drivers to respond calmly even if it is a quite unexpected situation. Very drivers suddenly brake. In the roadworks the obstacle is not visible (as it is supposed to be in the experiments), the warning allow he driver to slow down and change lane in time, the average deceleration is the lowest along the use cases.

Non-Equipped vehicles results

The equipped vehicles represents the best case for the system infrastructure in terms of communication and sensing, the positio was indeed provided by the vehicles and the strips did not play an active role in the warning system. The non-equipped are a hardest challenge for the project since they are connected through LTE and they rely on the strips to update their state in the Co-Driver instance. The state is not continously updated as it was for the equipped vehicles. The smartphone sensors are used only for logging purposes and to evaluate the performance. In principle, they could be employed by the Co-Driver, but here, the application for non-equipped vehicles are tested using only the minimal set of informations (in aurban environment the smartphone's GPS is less reliable than in a field experiment). The variability in general is higher than the one in the equipped vehicles. Figure 2.26 shows such variability in the width of the boxes. Non-equipped vehicles update the vehicle position to the Co-Driver only when they pass over a strip. Futhermore, a larger delay is introduced when it comes to communication. This two factors lie to the variability of the TTC. The warning evaluation cannot be updated continuously. Nevertheless, two third quantiles of TTC are well above the threshold for all applications, except VRU. Results are promising since the upcoming 5G technology may resolve the issue of the delays and for the positioning we can increase the number of the strips.

Time to collision for non-equipped vehicles

As discussed above, the TTC values are more variable and in general lower than those of the equipped vehicles. This is due to asynchronous update of the state fo the vehicle (velocity and position) and to the delay of communication. For readers that are interested in the communication-related results the reference is [5]. With non-equipped vehicles, the Co-Driver cannot rely on a continuous update of the vehicle state and furthermore the acceleration value is not provided by the strips. The position and the velocity are known to the system only for stips passages. Therefore, the warning, instead to be issued as soon as the Co-Driver would detect a dangerous intention form the state of the vehicle, in the non-equipped case, we need to wait for the next strip. One can notice that the threshold of 3 seconds is overcome in some experiments for the pedestrian use case (VRU in the figure), while in the other three applications it is not. This happens for a significant number of experiments: only the 60% percentile of the experiments is above the threshold. the VRU is the first experiment we make, after that we changed the layout of the strips (We had only few strip prototypes, in a real applications the strips would be more). In general we moved behind the strips, In the case of the intersection (INT), it was not necessary to modify the layout, since it was already more conservative than the other use cases for the reason we discussed for the equipped cases. Also with the compensation, some experiments has TTC below 3 seconds, a precise case is identified at which this happens: the first strip is passed at a safe velocity, but the second is not, the warning is therefore issued only at the second strip. This problem in real applications must be faced with a higher density of strips. In these experiments the number of strips (3 per lane) was limited by the availability of prototypes.

Average Acceleration for Non-equipped vehicles

The average acceleration for non-equipped vehicles is comparable with the equipped vehicles cases. The variability is still higher, but the average acceleration in general lies above the expected threshold. More outliers with high deceleration are present These cases correspond to the very lower values of the TTC (one by one is verified) the reason why the deceleration is high is the late warning.

Power two wheeled vehicles results

The PTW vehicles shows that the system is suitable for the PTW and for this hbrid configuration. Since both the TTC and the average acceleration are similar to the equipped cases. These results drives the attention to the layout of the strips, since they show that with LTE connection one can get similar performance to the equipped case if the state is continuously updated.

Conclusions

We developed Co-Driver based safety applications based on the SAFESTRIP “self-explaining” road technology. The four use cases are successfully covered for equipped and non-quipped vehicles even if some of them presented more promising results (wrong way driving and roadworks), the pedestrian protection was validated but it was the less performing use case for non equipped vehicles due to the layout that was not yet optimized for the experiments. Equipped vehicles and PTW “semi-equipped” vehicles behaves as expected, in particular the PTWs shows that the continous update of the state of the vehicle (and the acceleration) is important to reach high performance. The system performed well also for Non-equipped vehicles but some limits mainly due to strip prototypes shortness came out. A higher density of the strip is required to reach comparable performances w.r.t. the equipped vehicles. The Co-Driver was able to detect the in intention when fed with vehicle states since the overall average accelerations matched the expectancies for aware drivers.

Chapter 3

Model predictive control for hybrid electric vehicles

In this chapter, we deal with the problem of optimal online control to split the torques in the powertrains of hybrid electric vehicles, having as objective to minimize fuel consumption and battery capacity degradation (thus maximizing the battery's lifespan).
The overall problem is divided into two subproblems. The driver intention estimation for the prediction of the future velocity profile. The optimal torque distribution of the hybrid powertrain following a Model Predictive Control (MPC) scheme.
For the velocity prediction, the selected method is the Co-Driver presented in Section 3.2. Where it is explained, validated, and compared with other methods.
The optimal control problem addresses both fuel consumption and battery life preservation using an equivalent cost. In addition, the velocity profile is given. Therefore the performance of the vehicle is not affected. The battery aging (also called "battery degradation") is modeled through a state-of-the-art electrochemical model that returns the loss of capacity. The Energy Management System (EMS) is evaluated (comprehensive of the Co-Driver predictor) using actual driving data divided into 30 driving records from 10 drivers characterized by different driving styles. The dataset is presented in 3.2.1, the same dataset is used for the validation of the Co-Driver predictor. The simulations show that the proposed EMS can significantly reduce the capacity loss of the battery while preserving the fuel-saving performances, and the Co-Driver-based predictor is suitable to feed such EMS. A comparison of the EMS performances is carried out on different benchmarks. Firstly, we compare our proposed framework with an offline optimization performed on the entire dataset length. Secondly, we apply the optimal control technique proposed in an analogous receding horizon approach but using an ideal (non-causal) prediction using "future" data of the velocity. Finally, to investigate the contribution of the battery aging, all the benchmarks and the proposed framework are repeated, minimizing only the fuel consumption leaving the battery-free to degrade. This benchmark is repeated with two different horizon lengths. The couple Co-Driver and MPC approach show similar performances with the aforementioned ideal benchmarks.

3.1 Introduction

Hybrid Electric Vehicle (HEVs) gained popularity in the last decade because the impact of mobility on the emissions of greenhouse gases is still considerable, and such vehicles represent a step forward in reducing mobility impact on the environment. Electric motors can recover kinetic energy during the braking phase of driving that would be otherwise dispersed in heat.

One of the characteristics of the HEVs that can be exploited is the redundancy of the power output. The power can come from the electric part of the powertrain or the internal combustion engine (ICE) according to the necessities. The motor and the ICE present different characteristics in terms of efficiency that vary on the exerted torque and shafts' speed. Thus the torque request can be distributed between the two sources to optimize quantities (i.g. efficiency). The family of control strategies that address this type of problem is called Energy Management Sistem (EMS). The EMS optimizes some merit quantity making sure that the torque that the driver requests is provided instant by instant. EMS can use the current power request with a classic reactive control scheme. However, the most sophisticated systems take advantage of the knowledge of the future torque request, and have higher capabilities to improve the performance of the powertrain [50]. Thus, control strategies for EMS that are based on future knowledge and driver's intention estimation have gained research interest. In this chapter,, we address the on-linee optimization of the torque distribution for hybrid electric vehicles with two concurrent objectives: minimizing fuel consumption and minimizing battery aging.
The literature on the control strategies for HEV is wide and varied; In [132, 19, 115, 114] recent exhaustive surveys can be found.
A first broad classification can be made on the main logic they are based on. Firstly, we can consider the strategies that rely on a predefined logic to minimize some target quantities instant by instant. We can call them Rule-Based (RB) strategies. Secondly, we can consider the methods that exploit the possibility to directly minimize some target quantity (e.g., fuel consumption) using optimization techniques (i.g. optimal control). Our proposed approach falls under this category.
Between these two categories, the RB strategies still represent the most straightforward solutions for the commercial hybrid vehicles to equip them with real-time controls [114]. Such type of logic, ,usually requires little computational power, and it can benefit from the experience of the designers [137], making their implementation and testing easier concerning the optimization-based strategies.
Despite these pros of the RB, they present some cons. Their design process requires a long phase of tuning the parameters and, most importantly, when it comes to performance, optimization strategies outperform the RB, [78],[137].
Sophisticated rule-based strategies are often used in couple with optimization strategies. For example, in [65] the author's combined rules with an ECMS framework. They compared the results of their approach with the solution of an off-line dynamic-programming optimal control problem.
The common characteristic of all the Optimization-based approaches is the direct minimization of some cost functions. These objective functions usually comprehend the fuel consumption, and it may include some other quantities like the battery degradation. Despite this common goal, current literature presents different techniques. The most popular techniques are: the model predictive control (MPC) technique, the so-called Equivalent Consumption Minimization Strategie (ECMS), and offline optimizations. The ECMS method is widely spread in the literature usually, it concerns specifically the minimization of fuel consumption. It makes use of the Pontryagin Minimum Principle (PMP) to satisfy the first-order optimality condition and find the minimum of the fuel consumption.
The canonical ECMS cost function is composed of a term with the fuel consumption and a term with the power flow of the battery, both inflow and outflow depending on the sign. The name "equivalent consumption" is due to converting the power flow of the battery into equivalent fuel consumption. The weight that multiplies the battery power in the cost function has this scope, and it is called "equivalence factor" [92].
Recent realizations of this strategies can be found in [137] and [97]. In [137] the authors extended the classic ECMS strategy by considering the degradation of the battery, adding an empirical model of such degradation that depends on the current throughput. In [106] the battery degradation is considered using a look-up table that associates the instantaneous state of the battery and the current flow to various rates of battery degradation. Another recent instance is [82], in which the classic ECMS a map of the equivalence factors that depend on the driving cycle, such map is tuned using machine learning techniques.
The ECMS strategies can be employed as an fast optimization algorithms. Thus, they do not require knowledge of the future power demand unless they incorporate some method to change the equivalence factor on-lined on estimations of future power demand.
In [135] the authors propose an energy management strategy based on the ECMS framework that is adaptive. The framework is designed to optimize both flexible torque requests, and the gear shift commands.
Even if they cannot be used for real-time applications, several off-line optimal control techniques can be found in literature[101][123] since they can provide a benchmark to design and tune the rules for the RB methods.
When it comes to apply optimal control (or optimization in general) online , we can talk about Model Predictive Control (MPC) or Receding Horizon strategies. These methods require knowledge of future power demand (as

the word "predictive" suggests). Model Predictive Control, in short, is composed of a model of the system, a cost function to minimize, and knowledge (or an estimation) of a finite horizon of the future desired outputs. At every execution step, optimal control techniques are used with a system model, trying to minimize the cost function without affecting the desired outputs. Some general treatment of the MPC paradigm can be found in [51],[9] and [27].
The main MPC limitation concerns the real-time computational feasibility. The optimal control techniques used in such approaches usually have a high computational load.
Recent work is [130], where the authors use an MPC technique to minimize simultaneously fuel consumption, battery degradation, and electric energy consumption in an equivalent cost function. The overall model comprehends a simple equivalent-circuit battery model and a semi-empirical capacity degradation model. The low complexity of the models favors the usage of on-line dynamic programming a the expense of modeling accuracy. The future power demand is retrieved from an estimation of the future desired speed profile.
An on-line dynamic programming approach is similarly used in [136] to minimize fuel consumption. The authors propose a hierarchical velocity prediction strategy to estimate the future velocity profile. In [129] dynamic programming is combined with reinforcement learning. Another approach that features the multi-objective minimization of fuel consumption and battery degradation is presented in [79]. The speed profile is tracked as well as a target State-Of-Charge (SOC). The on-line optimal control problem is formulated with a direct multiple shooting method, and the resulting optimization problem is solved with sequential quadratic programming.
Even if we picked the most recent instances of sophisticated EMS that consider fuel consumption and battery aging, most of the approaches (and those implemented in commercial vehicles) still consideronly fuel consumption. However, it is well known that a severe battery usage leads to a degradation of its capacity and other performances. This eventually affects the vehicle maintenance costs and the environmental impact. The exhausted batteries may need to be replaced, and severe usage leads to a more frequent replacement. The frameworks above that account for the battery degradation model the batteries with equivalent circuits, which cannotodel the battery dynamics through all the range of the working conditions.
Sometimes more accurate models are taken into account, in [90] a state-of-the-art electrochemical model is used in an optimal control framework. But it is employed only to to update the control constraints considering the aging, but in the resolution of the optimal control problem,, the authors model the battery dynamics by using a more approximated model based on an equivalent circuit.
A motivation to use an accurate electrochemical model for the battery lies in the importance of its internal states. they are crucial to represent the degradation dynamics [108] accurately, [109]. In our EMS, we consider such a model in the resolution of the OCP. An off-line version of this idea was first introduced in [36]. In that work, the authors propose an off-line OCP with a state-of-the-art battery model. They validate the approach in simulation on some standard driving cycles.

In this chapter we extend the results of [36] from the off-line approach to the on-line. Here a similar OCP and the battery model validated in [36] are cast in a Non-linear Model Predictive Control (N-MPC) framework.
A detailed electrochemical battery model that is employed in an on-line optimization-based EMS is not present in the literature to the best of our knowledge.
This work presents an architecture for an energy management system of a hybrid electric vehicle. It is coupled with the Co-Driver prediction scheme seen in Section 3.2, making it a complete EMS.
The prediction is made using the vehicle's velocity and acceleration only. To summarize, we present an on-line optimization strategy that considers the the battery degradation model while being executable in real-time. The model used for the optimal control problem (OCP) is a state-of-the-art electrochemical battery model that includes the aging dynamics and the consequent capacity loss. The OCP minimizes the total cost: the price of the battery and the fuel price are used as weights to convert the cost function into an equivalent economic cost. The approach is validated on naturalistic driving telemetries (a dataset of 30 driving cycles performed by 10 drivers). The off-line approach proposed by some of the authors in [36] is reproduced on the real dataset, and it is used as a benchmark for the on-line approach.
The proposed EMS is first tested in pair with an exact velocity prediction (employing two different prediction time-horizon lengths) to validate the choice of the prediction horizon length. Then, the EMS is tested in pair with the novel prediction scheme to assess the prediction's effectiveness in feeding the driver's future intention to the EMS. This chapter presents a complete architecture for an energy management system of a hybrid electric vehicle. It combines two elements: Firstly, the bio-inspired the Co-Driver prediction algorithm explained in 3.2, which uses the vehicle's velocity and acceleration only. Secondly, an on-line optimization strategy that considers battery aging issues while being executable in real-time. The model used for the optimal control problem (OCP) is

a state-of-the-art electrochemical battery model that includes the aging dynamics and the consequent capacity loss. The OCP minimizes the total cost: the price of the battery and the price of the fuel are used as weights in order to convert the cost function into an equivalent economic cost. The approach is validated on naturalistic driving telemetries (a dataset of 30 driving cycles performed by 10 drivers). The off-line approach proposed by some of the authors in [36] is reproduced on the real dataset, and it is used as a benchmark for the on-line approach. The proposed EMS is firstly tested in pair with an exact velocity prediction (employing two different prediction time-horizon lengths) in order to validate the choice of the prediction horizon length. Then, the EMS is tested in pair with the novel prediction scheme to assess the effectiveness of the prediction in feeding the future intention of the driver to the EMS.
To prediction that feeds the future velocity of the EMS presented in this chapter is based on the Co-Driver concept in chapter 1. The Co-Driver instance used for the predictor is detailed in 3.2 and the dataset is detailed in Section 3.2.1. This chapter contains a Section 3.3 in which the powertrain and battery model used in the EMS are reported. The EMS is described in Section 3.4 in terms of the optimal control problems formulation and summarized in 3.5. In Section 3.5.3 the validation methodology is discussed and finally the results are discussed in 3.6, as well as the drawn conclusions.

3.2 Co-Driver based speed prediction

Obtaining an accurate estimation on a long time horizon is a challenging task since the speed profile is affected by many factors, including the driver's possibility to change his/her short-term goals, maneuver, or the urgency to adapt to unexpected situations. In this work, the time horizon of the speed profile prediction is the short-term one (up to 10 seconds), aiming to reach relatively high accuracy. Behind this objective, there is the hypothesis of the driver to be following one intention only and predicting simultaneously the intention and the velocity profile that will generate from it. The first thing to present now is the dataset.

3.2.1 Dataset

In this section, we introduce the experimental data used to validate the prediction scheme and the energy management system proposed in this chapter. The dataset comes from the user tests of the EU FP7 "interactIVe" project [3]. In the user tests, the rate of the logging is 10Hz. We are interested in longitudinal velocity and acceleration. Other sensors outputs were logged, some of them at a higher sampling rate, but we do not use them in this work. The user tests took place around the city of Turin in Italy; the route consisted mainly of extra-urban roads, but it included intersections, roundabouts, and some motorways segments. The map of the entire route is shown in Figure 3.1. A total of 10 different drivers drove the route. In order to get a proper dataset for our purposes, we must cut the time series, and we need to remove some parts.

Data preprocessing

The 10 datasets of driving are about 45 minutes long. As one can notice in Figure 3.1, a significant portion of the circuit consists of a motorway, the A55. We decided to discard such portion of the dataset for two reasons:

- The constant velocity would improve the prediction results since constant velocity is a trivial case for the predictor.
- The potential fuel saving decrease drastically when the velocity is constant since the vehicle is not a plug-in hybrid. It is common knowledge that most of the fuel-saving in an HEV vehicle occurs only in the presence of accelerations (positive and negative), since the battery can save the vehicle's kinetic energy in the form of chemical energy.

Another processing that we made concerns a division of the dataset. In order to tune the parameters of the predictor, we must have a dataset for tuning and one for validation. Both of these datasets must cover all the drivers. Furthermore, to make the comparison between results easier, it is desirable to have driving records of the same length. To meet these requirements, we take advantage of the similarity between the datasets. Looking at

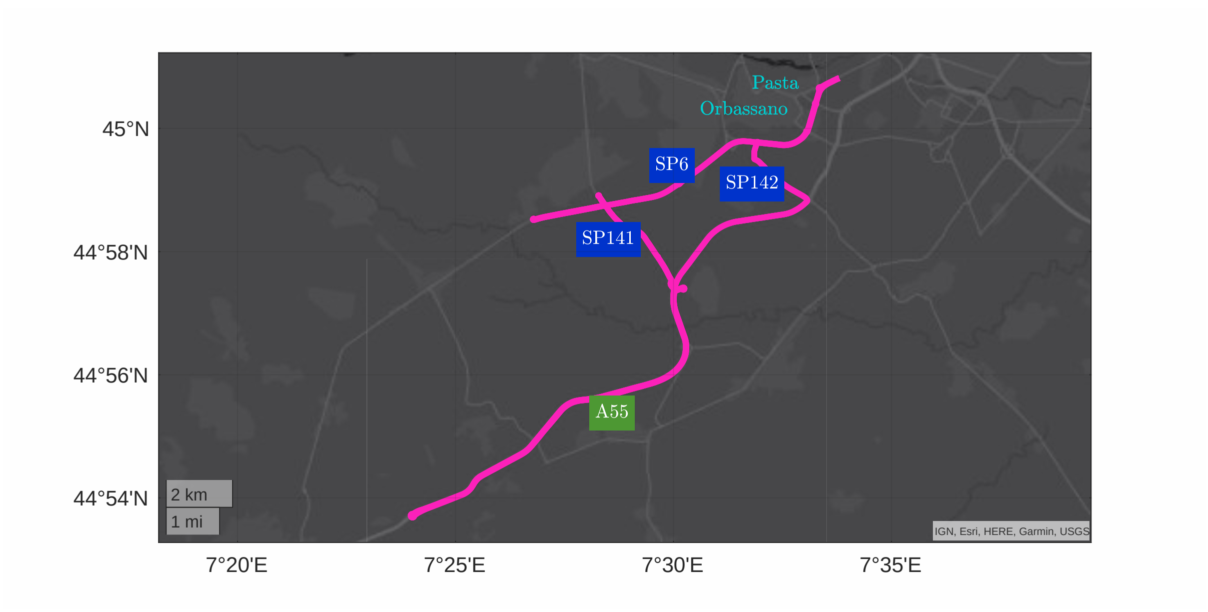


Figure 3.1: Map of the route of the user tests of the EU FP7 "interactIVe" project [3].

Figure 3.2, one can easily recognize a familiar pattern: one initial phase with varying velocity, one-second phase with high and constant velocity, and finally another but shorter phase with varying velocity. In Figure 3.2 the velocity ranges between the ten different datasets is shown. For every value of time, the maximum, the minimum, and the velocity are depicted. As shown in Figure 3.2, for each driving dataset, we cut three different records, namely A,B, and C. Dataset A starts 50 seconds after the beginning of the dataset and stops after 360 seconds. Dataset B and C are adjacent in time, covering $t_{\mathrm{C}} = [1840, 2200]$ and $t_{\mathrm{B}} = [1480, 1840]$.
Once we have 3 driving records for each driver, we have a total of 30 records. These data serve for tuning and validation of the prediction parameters.
We hereby define two different datasets that serve for the prediction algorithm validation. The predictor presents a two-parameter to be tuned; to validate the tuning of such parameters, we need two datasets: one for tuning and one for validation. In order to span all the drivers, the two datasets are composed as follows:

Tuning dataset For each driver, we pick **2 out of 3** A,B,C record randomly.

Validation dataset For each driver, The only record left that is not used in the tuning dataset is used for the validation dataset.

3.2.2 Prediction scheme

In this chapter, we propose an architecture for the prediction that is based on the Co-Driver introduced in 1. To infer the driver behavior, we use a generative "mirroring" approach discussed in the same chapter. Different instances of motion primitives are generated, and only one of them is selected through a Bio-inspired action-selection process: the Multi Hypothesis Sequential Ratio Test (MSPRT) [100][13]. This time we have in mind a precise scope for the predictor: it must have the same performance among different types of drivers without the necessity of tuning parameters. Therefore the maneuvers should be unavoidably a trade-off between accuracy and generality. In Figure 3.3, a high-level scheme of the predictor is shown. The predictor is based on a simple Co-Driver factoring two elements. Firstly, one action-priming block generates the motion primitives, which are the competing affordances. Secondly, one action-selection block that selects one primitive among the competing primitives generates by the previous block.
All the operations are executed with a receding horizon method, meaning that the action-priming block generates the three maneuvers at every time step, and the action selection block selects one of them according to new and previous evidence. We observed that the computational load of the prediction is negligible compared to the optimal control one. The data were available every 100ms, this time interval is not enough for the optimization step to

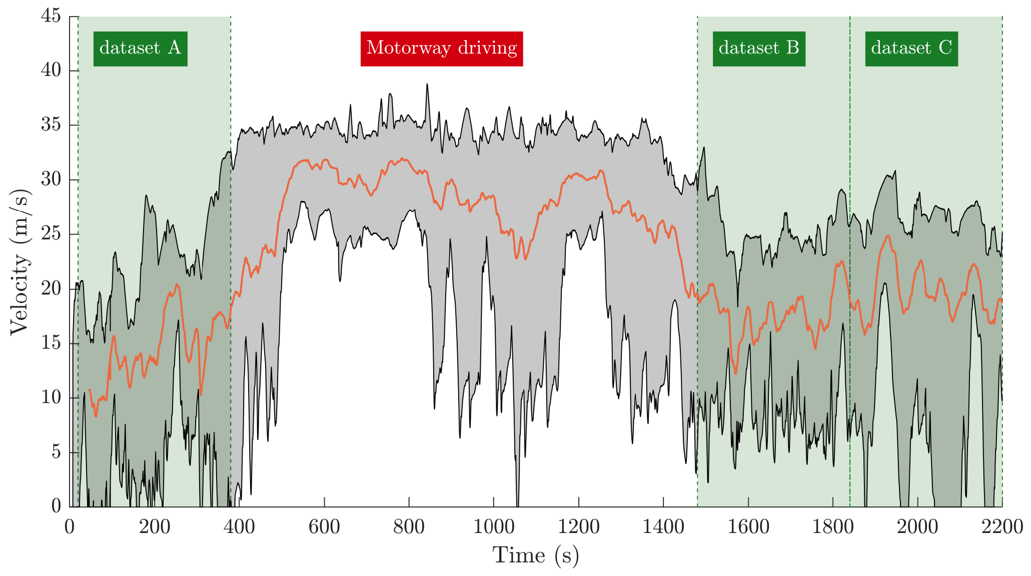


Figure 3.2: dataset cut for each driver, one record is cut into three sub-records, and the motorway driving is discarded. In orange, the average velocity among the datasets at each time, the range between the minimum and the maximum in grey.

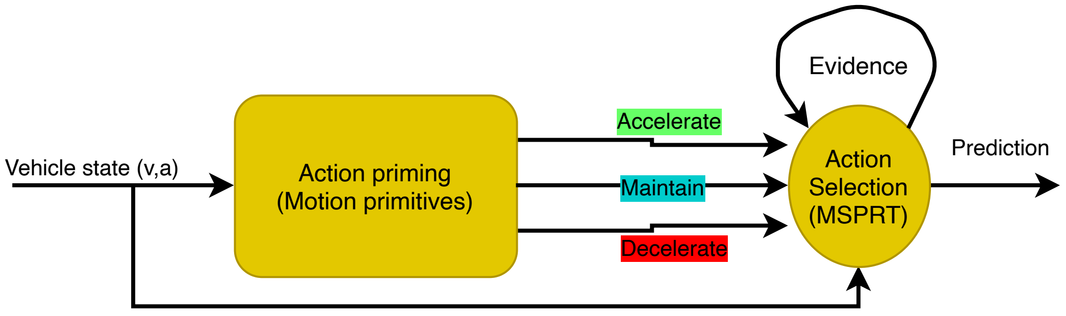


Figure 3.3: Predictor scheme, the affordances are indicated with their associated maneuvers.

finish, but the predictor can be executed. Thanks to this low computational load, we execute the predictor at 10 Hz, 5 times faster than the optimization. This higher rate allows the MSPRT to collect more samples per second and to cumulate more evidence, speeding up the update of the action selection. More details will be provided in the MSPRT Section 3.2.4. The following sections explain the predictor's single elements (blocks) in detail.

3.2.3 Action-Priming

At every time step, the action-priming block generates a new set of competing primitives. It gets the current "longitudinal" state of the vehicle $[v(t_\mathrm{i})\ ,\ a(t_\mathrm{i})]'$ as input at time t and it carries out three types of possible future action (primitives):

- **Accelerate**
- **Decelerate**
- **Maintain**

Every action type corresponds to one primitive with different final conditions. They are explained in the following sections. However, we need to define some quantities before moving to the description of the actions. The first thing to point out is that the duration of the maneuver T_{man} does not necessarily coincide with the prediction horizon time T, which can be shorter. Thus, we define T is the time horizon of the prediction and T_{man} for the actual motion primitive duration. The horizon time T in this predictor is equal to $5s$.

Accelerate

This type of action models when a driver is going to increase his/her velocity. For instance, changing to a higher velocity lane or driving a highway ramp. The FFCT primitive Free Flow Constrained Time is chosen, and it consists in a free flow with a target velocity and constrained time (See Section 1.3.4) We estimate the final target velocity v_{T} using the current acceleration value. We use a constant acceleration assumption only to compute a guess of v_{T}. Once the final velocity is established, the more realistic motion primitive is used to pursue that velocity target v_{T}. The current acceleration is preferred to the current jerk. Indeed the jerk is very difficult to retrieve, as stated in [17]. The primitive instantiated for the acceleration is the following:

$$M_{\mathrm{acc}} = \mathcal{M}_{\mathrm{FFCT}} \left(T_{\mathrm{man}}, 0, v_0 + a_0 T_{\mathrm{man}} \right). \tag{3.1}$$

For accelerate the duration of the primitive T_{man} is chosen equal to 10s. The choice of this parameter is the result of manual tuning.

Decelerate

This second action is the dual of the accelerated action. It models a driver who wants to stop or significantly reduce vehicle velocity. For the action to stop, following the minimum jerk concept, the literature provides more sophisticated models based on motion primitives. The work in [35] provides a pretty accurate model that divides the maneuver into phases. However, this accurate modeling requires more parameters that may change among drivers, and therefore, they do not fit our generality requirements. Therefore, we model the “so-called” main phase of the stop maneuvers (see [35]), which covers most of the duration of a stop procedure. The chosen maneuver is the same as in the acceleration action. The only difference lies in an additional constraint: the velocity must obviously be non-negative.

$$M_{\mathrm{dec}} = \mathcal{M}_{\mathrm{FFCT}} \left(T_{\mathrm{man}}, 0, \max \left(v_0 + a_0 T_{\mathrm{man}}, 0 \right) \right). \tag{3.2}$$

If one generates a decelerate primitive, it cannot accelerate one and vice versa. The final velocity depends indeed on the acceleration sign. Even if only one is generated, both are taken into account in the action-selection process (the missing one gets null evidence in the MSPRT selection mechanism) since the action selection process also depends on the past.

Maintain

This action models the small velocity fluctuations during a "cruise" state. One may be tempted to use a flat maneuver with a constant velocity profile in such a case. However, this solution would not account for minor (but not negligible) speed adjustments that frequently occur in driving. Therefore, we use a motion primitive that drives the vehicle smoothly from the current state to an estimated velocity adjustment. The final target velocity is again v_{T}, but it is expected to be very close to the current state v_0. This velocity is estimated again by selecting the FFCT motion primitive with the least instantaneous effort. The instantaneous effort is the initial jerk, therefore we select the maneuver that has $j_0 = 0$. The procedure to compute the final velocity v_f from the initial jerk $j(0)$ is provided in the FFCT dedicated Section 1.3.4 (Equation (1.11)). Note that the jerk is zero only for $t = 0$, thus it is not a constant acceleration maneuver. Once this velocity is known, the maneuver is defined and $T_{\mathrm{man}} = T$.

$$M_{\mathrm{mnt}} = \mathcal{M}_{\mathrm{FFCT}} \left(T, 0, v_{\mathrm{T}} \right). \tag{3.3}$$

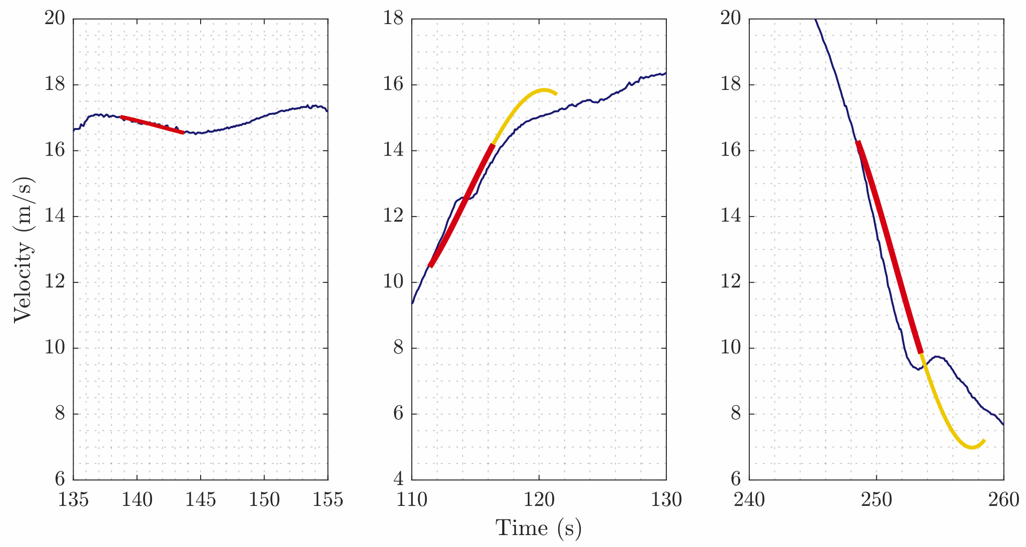


Figure 3.4: Example of three predictions. In blue the actual velocity, in red the prediction and the complete primitive in yellow.

Instances of prediction for each action type are provided in Figure (3.4). The selected actions are maintenance, acceleration, and deceleration from left to right. For each plot, the yellow line shows all the primitive; the red line represents the portion of the primitive in the prediction time horizon, which is the one used for prediction. At the same time, the blue line is the true velocity (from data).

3.2.4 Action-selection: MSPRT

The selection among the pool of actions provided by the action-priming block is made in the action-selection block using the MSPRT algorithm [13].One example of application in the automotive field can be found in [100]. This algorithm resembles the action-selection process that occurs in the basal ganglia of the vertebrates brain [57]. In this context, the Multi Hyphothesis Sequential Ratio Test is based on collecting evidence on which of the three potential actions the driver is pursuing (the action is also called *channel*). The key idea is to accumulate evidence at every time step for each channel (the accumulated evidence is called *salience*). One channel is then selected after the accumulated evidence is considered enough. Here the evidence metric ea_k is the reciprocal of the mean square error of the acceleration in a time window of the previous 0.5 seconds. The error, in this case, is the difference between the actual acceleration and the values given by the candidate maneuvers of a passed prediction step (the step occurred 0.5 seconds before the current time t_i). Given the index of the different channels $\mathrm{k} = 1, 2, 3$, The evidence is expressed by:

$$ea_{\mathrm{k,i}} = \frac{\sqrt{5}}{\sqrt{\sum_{\mathrm{p}=0}^{4} \left((a(t_{\mathrm{i}} - 0.5 + p\,t_{\mathrm{s}}) - \tilde{a}_k(t_{\mathrm{i}} - 0.5, p\,t_{\mathrm{s}}) \right)^2}}. \tag{3.4}$$

where t_{i} is the current time, t_{s} is the sampling time of the prediction (0.1 seconds). The values of the true acceleration are $a(t_{\mathrm{i}} - 0.5 + p\,t_{\mathrm{s}})$, which are known since are in the past up to the current value. The acceleration values $\tilde{a}_k(t_{\mathrm{i}} - 0.5, pt_{\mathrm{s}})$ are instead drawn from the three action candidates of 0.5 seconds before t_i. Past candidates are used instead of current candidates, since the motion primitives cannot be evaluated in $t < 0$, because such values are out of the maneuvers time domain $t \in [0, T_{\mathrm{m}}]$. For the computation of $ea_{k,i}$ we therefore require a set of past maneuvers that can be obtained in two ways indifferently. Past maneuvers can be stored in memory, or they can be computed again by means of stored past vehicle states. The evidence $ea_{k,i}$ is accumulated into the salience, appending and summing past values of the evidence. The number of past evidence values used for the

Salience is defined as w. This buffer length w is limited to 10 in order to avoid using outdated data. The salience is then defined as the mean of the evidence along a past window containing a number of samples equal to w:

$$Y_{\mathrm{k,i}} = \frac{1}{w} \sum_{j=0}^{w-1} ea_{\mathrm{k,(i-j)}}. \tag{3.5}$$

The length of the window w along which Y_{k} is initialized to 1, and it is incremented by one at every step, up to 10. If one action is selected, w is reset to one. Finally, the quantity used to select a channel is the *likelihood* L_{k} of each channel. It takes into account the salience of each channel in relation to the salience of the others. The likelihood assumes values from 0 to 1, $L_{\mathrm{k}} \in [0, 1]$ and the sum of the likelihoods of the channels is always equal to 1, $\sum_{k=1}^{3} L_{\mathrm{k}} = 1$. It reads as:

$$L_{\mathrm{k}} = \exp\left(Y_{\mathrm{k,i}} - \log \sum_{\mathrm{k}=1}^{3} \exp\left(Y_{\mathrm{k,i}}\right)\right) \tag{3.6}$$

One channel is selected if its likelihood reaches a predefined threshold L_{th}. If more than one channel reaches the threshold, the one with the higher likelihood is selected. Thus, the channel is selected according to:

$$k_i^\star = \mathsf{argmax}_{\mathrm{k}} \left\{ L_{\mathrm{k,i}} \mid L_{\mathrm{k,i}} > L_{\mathrm{th}} \right\} \tag{3.7}$$

where $k_i^\star$ is the selected channel at time t_i. The complete procedure for the MSPRT is illustrated in Algorithm 4. Finally, the threshold L_{th} must be tuned: if the value is too low, the channel changes continuously, and more than one channel may reach the threshold simultaneously. Conversely, if the value is too high, the MSPRT algorithm may be stuck without choosing a channel since the channels' likelihoods never reach the threshold. Note that if the chosen threshold L_{th} is more than 0.5, then only one channel at a time can exceed the threshold, and the argmax operator in Equation (3.7) is not required.

Algorithm 4 MSPRT algorithm at time t_i

```
Input: Past candidiate maneuvers at $t_i - 0.5$,
    Past evidence $ea_{\mathrm{k,(i-j)}}$ with $k = 1..3$ and $j = 0..(w-1)$.
Output: Index of the selected maneuver $k_i^\star$
 1: Initialize $w = 1$
 2: while true do
 3:     for k in 1..3 do
 4:         compute $ea_{\mathrm{k,i}}$ according to Equation (3.4)
 5:         compute $Y_{\mathrm{k,i}}$ using $j = 1..w$ ( using a number of $w$ past values of $ea_k$ )( according to Equation (3.5))
 6:         compute log-likelihood $L_{\mathrm{k,i}}$ (Equation (3.6))
 7:     end for
 8:     if $\max(L_{\mathrm{k}}) > L_{\mathrm{th}}$ then
 9:         select action $k^\star$ (Equation (3.7)
10:         reset $w = 1$
11:     else
12:         $k_i^\star = k_{i-1}^\star$, (follow previous action)
13:         $w = \max(w + 1, 10)$
14:     end if
15:     return output $k_i^\star$
16: end while
```

T_{man}	L_{th}
10 s	0.5

Table 3.1: Parameters for the prediction algorithm.

3.2.5 Validation

The goal of the prediction scheme is to provide a suitable velocity profile for the Energy Management Sistem that do not depend on the driver. The selected metric for the performance of the prediction is the root mean square of the error on the velocity sampled along the time horizon is computed. It is computed comparing the predicted speed profile ($\hat{v}(t_i, \tau)$) with the actual speed sample by sample every the prediction. Such error is expressed by:

$$\mathrm{RMSE_v(t_i)} = \sqrt{\sum_{\mathrm{k=1}}^{\mathrm{L}} (\mathrm{v(t_i + kt_s) - \hat{v}(t_i, kt_s)})}. \tag{3.8}$$

Where $v(\cdot)$ is the actual velocity and t_{i} is the current time at which the error is computed. Note that τ is the time distance from the current time t_i of the prediction. Note that $\tilde{v}(t_{\mathrm{i}}, 0) = v(t_{\mathrm{i}})$ and L is the number of the samples of the predicted primitive, which is always $L = 50$ in this chapter.

Parameters tuning

To meet this requirement we can tune the two parameters of the algorithm: T_{man} and L_{th}. The first parameter is the acceleration/deceleration primitive duration (the time horizon is always the same instead). The second parameter is the threshold of the MSPRT algorithm 3.2.4. The selected parameters are reported in Table 3.1.
To tune these parameters, we employed a trial and error approach on the tuning dataset (See Section 3.2.1), until the RMSE errors (See Equation (3.8)) lay under 3 m/s (excluding the outliers) for all drivers. Once the errors lie under this threshold on the validation dataset datasets, we verify the same condition on the validation dataset. In Figure 3.5 all the boxplots of the errors are shown, in the upper chart the errors on the validation dataset, in the middle chart the validation dataset and in the bottom chart, all the errors for each driver are merged.
Some considerations can be draw looking at Figure 3.5Most of the predictions present minor errors; as required in the tuning phase, the errors lie under 3m/s (excluding the outliers). Concerning the outliers, there is a strong presence of them. They correspond to predictions with higher errors. These errors correspond to moments when the intentions of the driver change suddenly, and the MSPRT takes some samples to collect enough evidence of the new current action. This is especially true when there is a switch between acceleration/deceleration, maintaining actions, and vice-versa.
Instances of predictions on two driving records are shown in Figure 3.6 (called “hairy plots”), which help us to understand the outliers. The two driving records are picked from driver 8, which presents low errors, and driver 6, which presents the highest errors (See Section 3.5). In both velocities plots, we can recognize some peaks and some sharp valleys. At that point, the predictor struggles to fit the future velocity. Such cases, with predictions that diverge from the actual velocity profiles, are the outliers in the boxplots of Figure 3.5. One way to improve the performance of the prediction may be to introduce more specific affordances (as new actions) related to the environment. For instance, the presence of a stop has to generate a “stop” affordances at the stop point. Another possible affordance can be the presence of other vehicles on the same lane that may generate the affordance to follow the vehicle and the affordance to overtake it. Note that an affordance related to another vehicle may play two roles: to generate the affordance above with the corresponding primitive and to inhibit the affordances that collide with the leading vehicle. This would allow exploiting the concept of affordance better.

3.2.6 Comparison with other methods

We compare the proposed Bio-Inspired velocity prediction with the following two approaches:

- Exponential decreasing torque demand

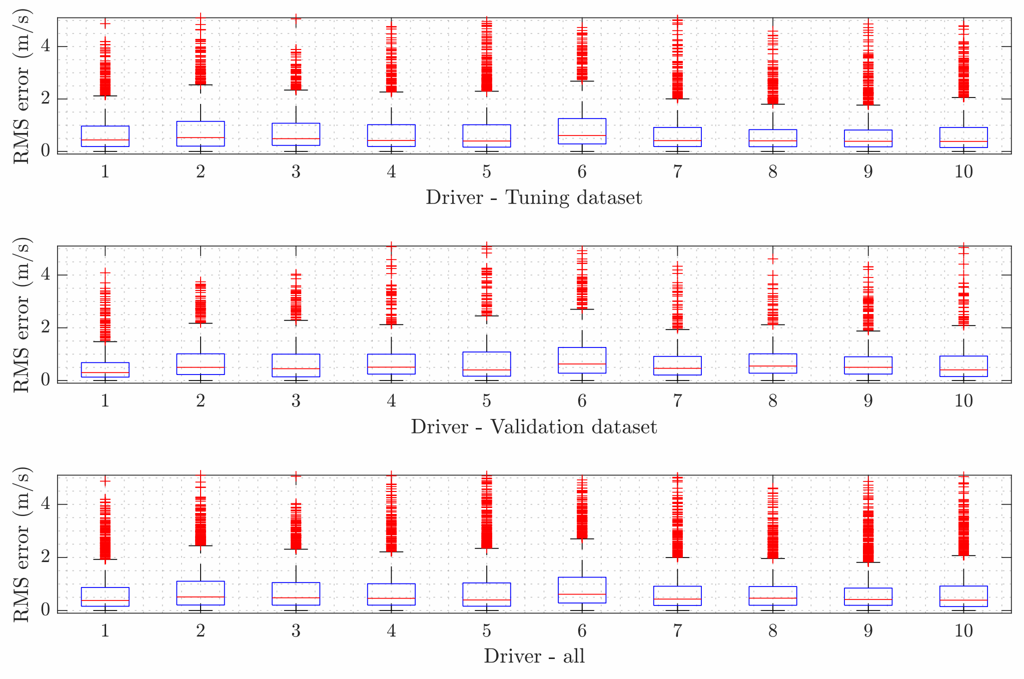


Figure 3.5: Box plot of the errors for each driver, tuning dataset, validation dataset and the total.

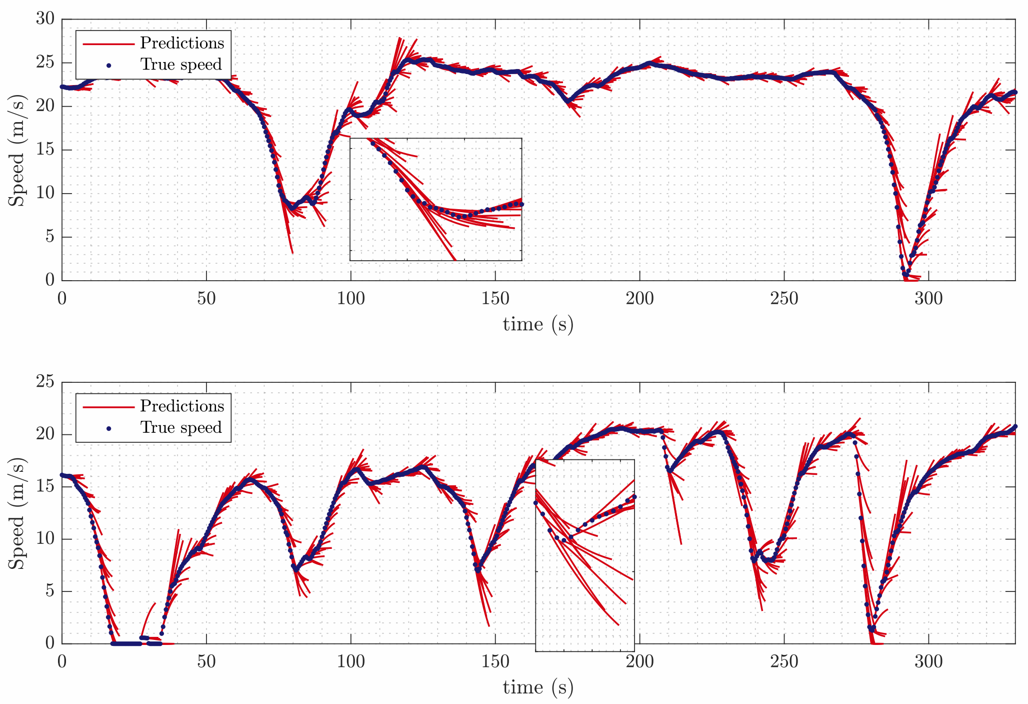


Figure 3.6: example of prediction: Hairy plots

- Markov chains

Such approaches are widely used in literature. All the methods are compared on a prediction horizon length equal to 5 seconds. The choice of this horizon length is motivated in Section 3.2.5. We hereby describe the methods and its usage in our work.

Exponential Decreasing Torque Demand

This method assumes that the driver torque request decreases exponentially, as described in [22][21]. The equation for the torque demand reads as follows:

$$T_{\mathrm{w}}(t_{\mathrm{i}}+kt_{\mathrm{s}}) = T_{\mathrm{w}}(t_{\mathrm{i}})\exp\left(\frac{-kt_{\mathrm{s}}}{\alpha_{\mathrm{d}}}\right), \;\; k=0,1,\cdots,L-1, \tag{3.9}$$

Where t_{i} is the starting time of the prediction horizon, L is the number of time samples in the prediction horizon, $T_{\mathrm{w}}(t_{\mathrm{i}})$ is the current torque request and α_{d} is the decaying rate of the torque request. We tune the α_{d} by solving an optimization problem, selecting the α_{d} that minimizes the RMS error between the predicted velocity and the dataset velocity. Once we estimated the torque request, the velocity is retrieved by the longitudinal dynamics in Equation (3.14).

Markov-Chain based Velocity Prediction

The use of discrete-time stochastic models in the form of Markov chains to predict velocity profiles of the driver (or related quantities) is widely present in the literature ([110, 134, 103, 107, 76, 89, 81]). We use a multivariate Markov chain to predict the future velocity and acceleration profiles. ([107, 110]). At first velocity and acceleration are discretized in M, N states:

$$\begin{aligned} v &\in \{\hat{v}_1, \hat{v}_2, \cdots, \hat{v}_M\} \\ a &\in \{\hat{a}_1, \hat{a}_2, \cdots, \hat{a}_N\} \end{aligned} \tag{3.10}$$

The *transition probability matrix* defines the Markov chain: in the case of the 1-Stage Markov chain the elements of the multi-dimensional transition probability matrix $\boldsymbol{T_{1S}} \in \mathbb{R}^{M\times N\times M\times N}$ are:

$$\begin{aligned} & t^{1S}_{(m,n),(p,q)} = \\ & P\left[v(t_{\mathrm{i}}+t_{\mathrm{s}}) = \hat{v}_p, a(t_{\mathrm{i}}+t_{\mathrm{s}}) = \hat{a}_q \,|\, v(t_{\mathrm{i}}) = \hat{v}_m, a(t_{\mathrm{i}}) = \hat{a}_n\right], \\ & m,p = 1,\cdots,M, \quad n,q = 1,\cdots,N, \end{aligned} \tag{3.11}$$

where $P[x]$ is the probability of the event x to happen, t_{i} is the current time instant and $(t_{\mathrm{i}}+t_{\mathrm{s}})$ is the next time instant. In the case of the 3-Stage Markov chain, two past velocity values are also considered as additional variables. ([134, 110]). the transition probability matrix $\boldsymbol{T_{3S}} \in \mathbb{R}^{M^3\times N\times M\times N}$ is defined as:

$$\begin{aligned} & t^{3S}_{(m_1,m_1,m_3,n),(p,q)} = \\ & P[v(t_{\mathrm{i}}+t_{\mathrm{s}}) = \hat{v}_p, a(t_{\mathrm{i}}+t_{\mathrm{s}}) = \hat{a}_q \,| \\ & v(t_{\mathrm{i}}-2t_{\mathrm{s}}) = \hat{v}_{m_1}, v(t_{\mathrm{i}}-t_{\mathrm{s}}) = \hat{v}_{m_2}, v(t_{\mathrm{i}}) = \hat{v}_{m_3}, \\ & a(t_{\mathrm{i}}) = \hat{a}_n], \\ & m_1,m_2,m_3,p = 1,\cdots,M, \quad n,q = 1,\cdots,N, \end{aligned} \tag{3.12}$$

Both the transition probability matrix is estimated from the actual driving dataset described in Section 3.2.1.

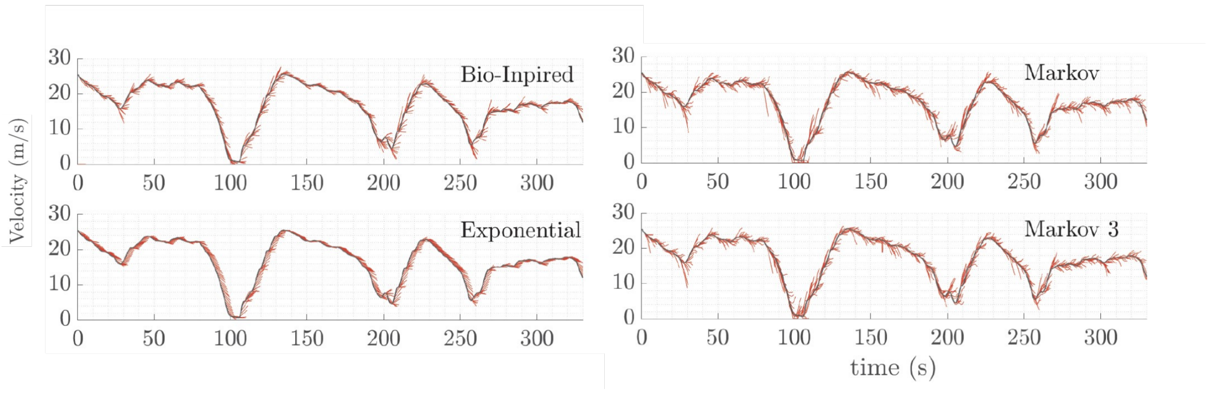


Figure 3.7: Example of outcomes of each velocity prediction method on a driving record from the naturalistic driving dataset.

Table 3.2: Average RMS error for Bio-inspired and Decreasing torque prediction methods for three validation dataset.

Validation	Bio-Inpsired $[m/s]$	M. 1-S $[m/s]$	M. 3-S $[m/s]$	E.Var.Torque $[m/s]$
RMSEv	0.74	0.77	0.81	0.89

Results of comparison on naturalistic driving dataset

The proposed benchmark methods are compared on our real driving dataset. Figure 3.7 shows the velocity prediction methods outcomes for one sample driving record of our dataset, as stated earlier, the horizon is 5 seconds. The root mean square of the velocity error (RMSE) is used as a metric of comparison. The metric (Equation (3.8)) is computed for each time step validation sets. The average RMSE is then calculated for the validation set. It is important to notice that for the Markov chains based methods the validation and the training set coincide, to avoid empty values in the transition matrices. The results are reported in Table The proposed Bio-inspired method performs better than the Exponential method, and even better than the Markov chains results on naturalistic driving set, which validation and the training sets are the same.
As a general result, we can conclude that our Bio-Inspired framework for the velocity prediction is more accurate, according to the RMSEv metric, than the other frameworks. The Exponential Decreasing torque method presents good performances when the accelerations (positive and negative) are small, but the proposed Bio-inspired approach can provide a better prediction in the presence of high and continuous acceleration values. Exponential Decreasing torque method strongly depend on the choice of its parameter α. The Markov chain-based methods can provide better performances regardless of the acceleration values, even in the presence of intention changes (for example switching from braking to accelerating) but only on the dataset, they are trained in. Additionally, the Markov chains require a large amount of memory and data. Our Bio-inspired framework provides maneuvers that are closer to real ones. Since it is based on a model of a human driver. Therefore, the Bio-inspired method represents an improvement in the literature, and it is computationally light. Furthermore, it does not require a large amount of data to be tuned since the two parameters involved (L_{th} and T_{m}) have a phisical meaning and can be tuned manually.

3.3 Powertrain and battery model

In this Section we briefly describe the models used for the implementation of the on-line EMS, in [36]: a parallel mild HEV. A scheme of the powertrain is shown in Figure 3.8. It includes an electric motor that is always connected to an internal combustion engine (ICE) shaft through a geared coupling. A clutch connects the ICE shaft an automatic transmission featuring eight speeds. The two exerted torques are $T_{\mathrm{eng}}(t)$ and $T_{\mathrm{mot}}(t)$, they are generated by the ICE and by the electric motor respectively. They sum up at the ICE shaft and are transmitted to the driving wheels undergoing the transmission ratio of the transmission and the differential. The total torque

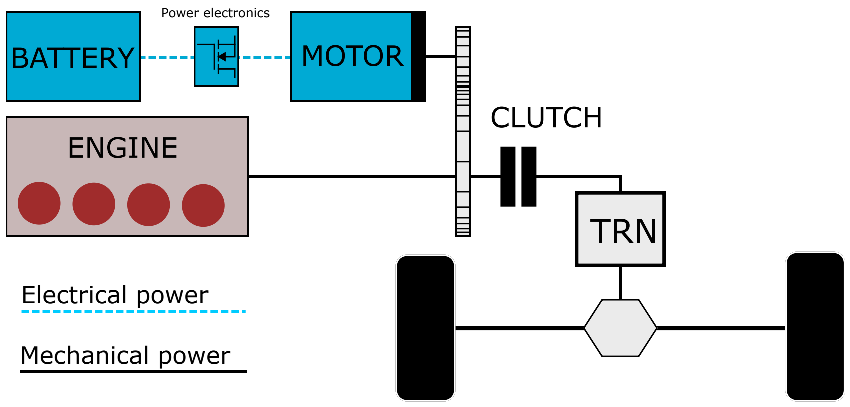


Figure 3.8: Parallel hybrid powertrain considered

$T_{\mathrm{w}}(t)$ the following expression gives at the driving wheels:

$$T_{\mathrm{w}}(t) = \Big(T_{\mathrm{eng}}(t) - J_{\mathrm{eng}}\,\dot{\omega}_{\mathrm{eng}}(t) + T_{\mathrm{mot}}(t)\gamma_{\mathrm{mot}}\Big)\,\eta_{\mathrm{trn}}^{\mathrm{sign}[a_{\mathrm{x}}(t)]}\,\gamma_{\mathrm{trn}}(t)\,\gamma_{\mathrm{axle}} + T_{\mathrm{brk}}(t) \tag{3.13}$$

3.3.1 Longitudinal Dynamics Model

The following longitudinal dynamics equation rules longitudinal velocity and acceleration and the total torque:

$$T_{\mathrm{w}}(t) = \big(c_{\mathrm{f}}\,v_{\mathrm{s}}(t)^2 + m_{\mathrm{v}}\,a_{\mathrm{s}}(t)\big)\,r_{\mathrm{w}} + (g\,m_{\mathrm{v}}\,(\sin(\sigma(t)) + c_{\mathrm{rr0}}\cos[\sigma(t)]\,))\,r_{\mathrm{w}}, \tag{3.14}$$

where v_{x} is the longitudinal speed of the vehicle, and σ_{road} is the road slope, which in this work is set to zero for data availability reasons. Table 3.3 shows the related parameters.
We are interested in modeling the ICE fuel consumption as a function of the speed and torque of the shaft. Willan's lines approach [99] consists in an affine function that approximates the fuel consumption rate $\dot{m}_{\mathrm{f}}$. The function has as arguments the engine mechanical power at the shaft, and it is expressed as follows:

$$\begin{aligned}\dot{m}_{\mathrm{f}}(\omega_{\mathrm{eng}}(t), T_{\mathrm{eng}}(t)) &= \alpha(\omega_{\mathrm{eng}}(t))\,P_{\mathrm{eng}} + \beta(\omega_{\mathrm{eng}}(t))\\ \alpha(\omega_{\mathrm{eng}}(\cdot)) &= \sum_{\mathrm{i}=0}^{3}\alpha_{\mathrm{i}}\,\omega_{\mathrm{eng}}^{\mathrm{i}-1}(\cdot)\\ \beta(\omega_{\mathrm{eng}}(\cdot)) &= \sum_{\mathrm{i}=0}^{3}\beta_{\mathrm{i}}\,\omega_{\mathrm{eng}}^{\mathrm{i}-1}(\cdot)\end{aligned} \quad . \tag{3.15}$$

The Willan's parameter values selected in this study are taken from [36]. The list of them is in Table 3.4.

3.3.2 Electric motor Model

The electric motor is alimented by the 48V battery and can act both as a motor and a generator depending on the sign of the torque. In this case, negative torque corresponds to the generator, therefore an inflow of power.

Description	Symbol	Value	Unit
Transmission efficiency	η_{trn}	0.95	-
Axle ratio	γ_{axle}	3.73	-
First Gear	$\gamma_{\text{trn,1}}$	5.00	-
Second Gear	$\gamma_{\text{trn,2}}$	3.20	-
Third Gear	$\gamma_{\text{trn,3}}$	2.14	-
Fourth Gear	$\gamma_{\text{trn,4}}$	1.72	-
Fifth Gear	$\gamma_{\text{trn,5}}$	1.31	-
Sixth Gear	$\gamma_{\text{trn,6}}$	1.00	-
Seventh Gear	$\gamma_{\text{trn,7}}$	0.82	-
Eighth Gear	$\gamma_{\text{trn,8}}$	0.64	-
Motor ratio	γ_{mot}	2.00	-
Vehicle mass	m_{v}	1750	kg
Dyn. wheel radius	r_{w}	0.31	m
Aerodyn. drag coeff.	c_{f}	0.28	$\text{kg}\,\text{m}^{-1}$
Rolling resistance coeff	c_{rr0}	0.02	-
Engine inertia	J_{eng}	$2.26 \cdot 10^{-2}$	$\text{kg}\,\text{m}^2$
Up-shift engine speed	$\omega_{\text{eng,up}}$	$4.77 \cdot 10^3$	rpm
Down-shift engine speed	$\omega_{\text{eng,down}}$	$2.87 \cdot 10^3$	rpm

Table 3.3: Vehicle and powertrain parameters.

	$i = 1$	$i = 2$	$i = 3$
α_{i}	$6.15 \cdot 10^{-8}$	$-5.24 \cdot 10^{-11}$	$7.47 \cdot 10^{-14}$
β_{i}	$7.04 \cdot 10^{-6}$	$8.33 \cdot 10^{-7}$	$6.30 \cdot 10^{-10}$

Table 3.4: Willan's coefficients [36].

The power flowing through the motor (in both directions) is a function of torque, speed, and motor efficiency $\eta_{\text{mot}}(\omega_{\text{mot}}(\cdot), T_{\text{mot}}(\cdot))$. The efficiency is usually provided by the manufacturer (as a chart), and it depends on torque and speed. The expression of the power P_{mot} is:

$$P_{\text{mot}}(\omega_{\text{mot}}(t), T_{\text{mot}}(t)) = \begin{cases} \omega_{\text{mot}}(t)\, T_{\text{mot}}(t)\, \eta_{\text{mot}}(t) & T_{\text{mot}}(t) < 0, \text{gener. mode} \\ \frac{\omega_{\text{mot}}(t)\, T_{\text{mot}}(t)}{\eta_{\text{mot}}(t)} & T_{\text{mot}}(t) \geq 0, \text{motor mode} \end{cases}. \tag{3.16}$$

The characteristic of the instantaneous power versus velocity and torque (including, therefore, the efficiency) is a smooth surface. It is convenient to map the power directly since the corresponding surface is smoother w.r.t the efficiency. A third-order polynomial fits this power characteristic:

$$P_{\text{mot}}(\omega_{\text{mot}}(t), T_{\text{mot}}(t)) = \sum_{\text{j}=0}^{3} \sum_{\text{i}=0}^{3} p_{\text{i,j}}\, \omega_{\text{mot}}(t)^{\text{j}}\, T_{\text{mot}}(t)^{\text{i}}. \tag{3.17}$$

3.3.3 Battery System

The model of the battery employed in this work is a state-of-the-art electrochemical model, detailed in [108] and [109]. The battery is composed of 180 gr/NMC lithium-ion 18650 cylindrical cells. A number of $n_{\text{p}} = 16$ parallel modules are present with $n_{\text{s}} = 13$ cell series for a total of 180 units. This is the same configuration of [36]. The nominal voltage of a cell is $3.6\,\text{V}$ and the nominal capacity is $2.0\,\text{Ah}$. The total (nominal) voltage of the battery pack is then $48\,\text{V}$, The energy stored is approximately $1.5\,\text{kWh}$. Maximum discharge and charge powers are between the values of $17.11\,\text{kW}$ and $11.5\,\text{kW}$ respectively; these values are derived from the maximum currents, 30 A and 20 A respectively. Therefore, the Equation of the total battery power is:

$$P_{\mathrm{batt}}(t) = V(t) n_{\mathrm{s}} I(t) n_{\mathrm{p}}, \tag{3.18}$$

where $I(t)$ is the current flowing through one cell and $I(t)$ is the voltage applied. We hereby make an assumption: the cooling system is able to maintain the temperature constant among the cells (it is indicated by T). In general, this does necessarily not hold for batteries, because high currents may lead to temperature gradients.

Charge/discharge dynamics

What makes the difference in battery usage is the charge and discharge dynamics. We hereby briefly recap the equations of such dynamics, presented in the electrochemical model in [36, 108, 109]. The Equation returns the voltage on a cell, and it is expressed as:

$$V(t) = (U^{\mathrm{p}}(t) - U^{\mathrm{n}}(t)) + (\eta^{\mathrm{p}}(t) - \eta^{\mathrm{n}}(t)) + (\phi_{\mathrm{e}}^{\mathrm{p}}(L,t) - \phi_{\mathrm{e}}^{\mathrm{n}}(0,t)) - R_{\Omega} I(t), \tag{3.19}$$

The input of the model is the current $I(t)$ applied to a single cell (galvanostatic mode). The terminal voltage $V(t)$ is the output and it is measured between the positive and the negative current collectors. It is composed of a sum of the potential and overpotential terms. The values of the model parameters are taken from [36].

We have the equilibrium potentials $U^{\mathrm{i}}(t)$ that depend on the concentration fn the lithium ion on the surface of the electrode particle (namely the solid-electrolyte interface). The concentration is indicated by $c_{\mathrm{s,e}}^{\mathrm{i}}(t)$. Given $\theta^{\mathrm{i}}(t) = c_{\mathrm{s,e}}^{\mathrm{i}}(t)/c_{\mathrm{s,max}}^{\mathrm{i}} \in [0;1]$ as the stoichiometry ratio the functional form of the equilibrium potential at the cathode side (see [133]) is given by:

$$U^{\mathrm{p}}(\theta^{\mathrm{p}}(t)) = -10.72(\theta^{\mathrm{p}}(t))^4 + 23.88(\theta^{\mathrm{p}}(t))^3 - 16.77(\theta^{\mathrm{p}}(t))^2 + 2.595\,\theta^{\mathrm{p}}(t) + 4.563 \tag{3.20}$$

while at the anode side (see [111]) is

$$\begin{aligned} U^{\mathrm{n}}(\theta^{\mathrm{n}}(t)) = 0.1493 + 0.8493\exp(-61.79\,\theta^{\mathrm{n}}(t)) \\ +0.3824\exp(-665.8\,\theta^{\mathrm{n}}(t)) \\ -\exp(39.42\,\theta^{\mathrm{n}}(t) - 41.92) \\ -0.03131\tan^{-1}(25.59\,\theta^{\mathrm{n}}(t) - 4.099) \\ -0.009434\tan^{-1}(32.49\,\theta^{\mathrm{n}}(t) - 15.74). \end{aligned} \tag{3.21}$$

The relation between the kinetic overpotential terms $\eta^i(t)$ and the current density $j^i(t) = \mp\frac{I(t)}{A^i L^i}$ is ruled by the Butler-Volmer equation. The solution is expressed as:

$$\eta^i(t) = \frac{R_{\mathrm{g}}T}{\alpha F}\sinh^{-1}\left(\frac{j^i(t)}{a_{\mathrm{s}}^i i_0^i(t)}\right), \qquad i = \mathrm{p}, \mathrm{n}. \tag{3.22}$$

The exchange current density $i_0^i(t)$, in turn, is related as well to the concentration at the electrode surface $c_{\mathrm{s,e}}^i(t)$

and to the concentration in the electrolyte $c_{\mathrm{e}}^{i}(t)$; The equation reads as:

$$i_0^i(t) = k^i \sqrt{c_{\mathrm{e}}^i(t)(c_{\mathrm{s,max}}^i - c_{\mathrm{s,e}}^i(t))c_{\mathrm{s,e}}^i(t)},\ i = \mathrm{p}, \mathrm{n}. \tag{3.23}$$

We can compute the electrolyte overpotential $\Delta\,\phi_{\mathrm{e}}(t) = \phi_{\mathrm{e}}^{\mathrm{p}}(L,t) - \phi_{\mathrm{e}}^{\mathrm{n}}(0,t)$ by integrating the equation of the conservation of charge in the liquid phase. Making the assumption constant current density throughout the electrodes, we can write the solution:

$$\Delta\,\phi_{\mathrm{e}}(t) = \kappa_{\mathrm{d}}(\log(c_{\mathrm{e}}^{\mathrm{p}}(t)) - \log(c_{\mathrm{e}}^{\mathrm{n}}(t))) - I(t)R_{\mathrm{e}}, \tag{3.24}$$

where $\kappa_{\mathrm{d}} = \frac{2R_{\mathrm{g}}T(1-t_0^+)}{F}(1+\beta)$. The electrolyte resistance is expressed by $R_{\mathrm{e}} = \frac{1}{\kappa\varepsilon_{\mathrm{e}}}\left(\frac{L^{\mathrm{p}}}{A^{\mathrm{p}}} + \frac{L^{\mathrm{n}}}{A^{\mathrm{n}}}\right)$, where κ is the electrolyte conductivity, which depends on temperature (constant among cells but variable in time). In [119] equations are detailed, presenting an experimental analysis of the electrochemical properties of a $\mathrm{LiPF_6}$-based electrolyte. The transfer function between the SOC and the surface concentration at the electrodes was presented in [36]. This model includes battery temperature variations; The state space expression is:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}(t)I(t), \quad \text{with } \boldsymbol{x} \in \mathbb{R}^7, \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t), \qquad \text{with } \boldsymbol{y}(t) = \begin{bmatrix} \tilde{c}_{\mathrm{s,e}}^{\mathrm{p}}(t) \\ \tilde{c}_{\mathrm{s,e}}^{\mathrm{n}}(t) \\ \widetilde{SOC}(t) \\ \tilde{c}_{\mathrm{e}}^{\mathrm{p}}(t) \\ \tilde{c}_{\mathrm{e}}^{\mathrm{n}}(t) \end{bmatrix} \in \mathbb{R}^5, \end{aligned} \tag{3.25}$$

where the tilde indicates a variation from the equilibrium. The formulation of the state-space matrices is reported in Section 3.3.4. The vector $\boldsymbol{x}(t) = [x_1, ..., x_7]^T$ contains the state variables, while $\boldsymbol{y}(t)$ is the output vector. The Arrhenius-like equation (3.26) provides the dependencies on temperature: The kinetic constants k^i, the solid phase diffusion coefficients D_{s}^i and the activity coefficient β increase with temperature, while R_Ω decreases with the it.

$$\Gamma(T) = \Gamma_{\mathrm{ref}} \exp\left(-\frac{E_{\mathrm{act},\Gamma}}{R_{\mathrm{g}}}\left(\frac{1}{T} - \frac{1}{T_{\mathrm{ref},\Gamma}}\right)\right). \tag{3.26}$$

3.3.4 Battery dynamics, State Space

We hereby report the matrices of the state-space realization. In Equations (3.25) In (3.27a), (3.27b), and (3.27c). We show only the nonzero elements.

$$
\begin{aligned}
\boldsymbol{A}_{1,1} &= -\frac{189 D_{\mathrm{s}}^{\mathrm{p}}}{[R_{\mathrm{s}}^{\mathrm{p}}]^2} \\
\boldsymbol{A}_{1,2} &= -\frac{3465 [D_{\mathrm{s}}^{\mathrm{p}}]^2}{[R_{\mathrm{s}}^{\mathrm{p}}]^4} \\
\boldsymbol{A}_{1,5} &= \frac{189 D_{\mathrm{s}}^{\mathrm{p}} (\theta_{100\%}^{\mathrm{p}} - \theta_{0\%}^{\mathrm{p}}) c_{\mathrm{s,max}}^{\mathrm{p}}}{[R_{\mathrm{s}}^{\mathrm{p}}]^2} \\
\boldsymbol{A}_{2,1} &= 1 \\
\boldsymbol{A}_{2,5} &= -(\theta_{100\%}^{\mathrm{p}} - \theta_{0\%}^{\mathrm{p}}) c_{\mathrm{s,max}}^{\mathrm{p}} \\
\boldsymbol{A}_{3,3} &= -\frac{189 D_{\mathrm{n}}^{\mathrm{n}}}{[R_{\mathrm{s}}^{\mathrm{n}}]^2} \\
\boldsymbol{A}_{3,4} &= -\frac{3465 [D_{\mathrm{s}}^{\mathrm{n}}]^2}{[R_{\mathrm{s}}^{\mathrm{n}}]^4} \\
\boldsymbol{A}_{3,5} &= -\frac{189 D_{\mathrm{s}}^{\mathrm{n}} (\theta_{100\%}^{\mathrm{n}} - \theta_{0\%}^{\mathrm{n}}) c_{\mathrm{s,max}}^{\mathrm{n}}}{[R_{\mathrm{s}}^{\mathrm{n}}]^2} \\
\boldsymbol{A}_{4,3} &= 1 \\
\boldsymbol{A}_{4,5} &= (\theta_{100\%}^{\mathrm{n}} - \theta_{0\%}^{\mathrm{n}}) c_{\mathrm{s,max}}^{\mathrm{n}} \\
\boldsymbol{A}_{6,6} &= -\frac{9.8710 D_{\mathrm{e}}^{\mathrm{eff}}}{L^2} \\
\boldsymbol{A}_{7,7} &= -\frac{9.5842 D_{\mathrm{e}}^{\mathrm{eff}}}{L^2}
\end{aligned} \tag{3.27a}
$$

$$
\begin{aligned}
\boldsymbol{B}_{1,1} &= \frac{7}{\varepsilon^{\mathrm{p}} F A^{\mathrm{p}} L^{\mathrm{p}}} \\
\boldsymbol{B}_{2,1} &= -\frac{[R_{\mathrm{s}}^{\mathrm{p}}]^2}{15\, D_{\mathrm{s}}^{\mathrm{p}} \varepsilon^{\mathrm{p}} F A^{\mathrm{p}} L^{\mathrm{p}}} \\
\boldsymbol{B}_{3,1} &= \frac{7}{\varepsilon^{\mathrm{n}} F A^{\mathrm{n}} L^{\mathrm{n}}} \\
\boldsymbol{B}_{4,1} &= -\frac{[R_{\mathrm{s}}^{\mathrm{n}}]^2}{15\, D_{\mathrm{s}}^{\mathrm{n}} \varepsilon^{\mathrm{n}} F A^{\mathrm{n}} L^{\mathrm{n}}} \\
\boldsymbol{B}_{5,1} &= -\frac{1}{Q_{\mathrm{N}}} \\
\boldsymbol{B}_{6,1} &= \frac{3.1463 (t_0^+ - 1)}{\varepsilon_{\mathrm{e}} L F A^{\mathrm{p}}} \\
\boldsymbol{B}_{7,1} &= -\frac{2.9351 (t_0^+ - 1)}{\varepsilon_{\mathrm{e}} L F A^{\mathrm{n}}}
\end{aligned} \tag{3.27b}
$$

$$
\begin{aligned}
\boldsymbol{C}_{1,1} &= 1 \\
\boldsymbol{C}_{2,3} &= -1 \\
\boldsymbol{C}_{3,5} &= 1 \\
\boldsymbol{C}_{4,6} &= 1 \\
\boldsymbol{C}_{5,7} &= 1
\end{aligned} \tag{3.27c}
$$

Battery degradation

The degradation (or aging) of the battery plays a fundamental role in the proposed EMS. It is not directly expressed in the State Space model of Section 3.3.4. Instead, it is included directly in the cost function of the optimal control problem. This is possible because the aging effect depends on the state only as an integrator. Therefore it is suitable to be used. To model the aging of the cylindrical $2.0\,\mathrm{Ah}$ cell, we adopt the model of the aging effect proposed in [75]. The equation for the solvent reduction kinetics (Butler-Volmer) is simplified as described in [36], it reads as:

$$j_{\mathrm{s}}(t) = -\frac{k_{\mathrm{SEI}}(T)}{2A_{\mathrm{n}}(1+\lambda\theta(t))\sqrt{t}}, \tag{3.28}$$

where $\theta(t) = \exp\left[\frac{F}{RT}\left(\eta^{\mathrm{n}}(t) + U^{\mathrm{n}}(t) - U_{\mathrm{sei}}\right)\right]$. The Arrhenius dependency in (3.26) depicts the behaviour of the kinetic coefficient for the side reaction $k_{\mathrm{SEI}}(T)$. The effect of the anode potential on the SEI growth is accounted by the fitting parameter λ (see [36]). The integral of the side-reaction rate over time is equal to the capacity loss associated to the SEI formation:

$$Q_{\mathrm{SEI}}(t) = \int_0^t j_{\mathrm{s}}(t)\, A_{\mathrm{n}} \mathrm{d}t\,. \tag{3.29}$$

Another aging factor is treated in [75]. The authors describe the increased capacity loss observed after discharging and charging cycles as due to the structural damages that constantly isolate the active material. The phenomenon can be described by the variation of the active material (Assuming a uniform utilization). It is expressed by:

$$\frac{\mathrm{d}\varepsilon_{\mathrm{AM}}(t)}{\mathrm{d}t} = -\kappa_{\varepsilon}(T)|j_{\mathrm{n}}(t)|. \tag{3.30}$$

where $\kappa_{\varepsilon}(T)$ is related to the temperature again according to (3.26). The SOC-dependent rate of capacity loss induced by the reduction of the volume fraction is expressed by:

$$\frac{\mathrm{d}Q_{\mathrm{AM}}(t)}{\mathrm{d}t} = \frac{\mathrm{d}\varepsilon_{\mathrm{AM}}(t)}{\mathrm{d}t} SOC(t)\, V^{\mathrm{n}}\, c^{\mathrm{n}}_{\mathrm{s,max}}. \tag{3.31}$$

The two capacity loss mechanisms are now combined, assuming the superposition of effects, providing the total capacity t. The final expression is then:

$$\begin{aligned} Q_{\mathrm{loss}}(t) &= Q_{\mathrm{SEI}}(t) + Q_{\mathrm{AM}}(t) \\ &= -\int_0^t \frac{k_{\mathrm{SEI}}(T)}{2(1+\lambda\theta(t))\sqrt{t}}\mathrm{dt} \\ &\quad -\int_0^t \kappa_{\mathrm{AM}}(T)\, SOC(t)\, |I(t)| \mathrm{d}t, \end{aligned} \quad . \tag{3.32}$$

The constants in (3.31) are condensed in the equivalent parameter $\kappa_{\mathrm{AM}}(T)$. From the last term in (3.32) we caneasily see that high currents at high SOC states will lead to higher aging rate of the battery.

3.4 Energy Management System

The scope of the EMS is to compute instant by instant the torque split between the electric motor and the ICE that minimizes the total cost of the fuel and the battery capacity loss. The term "cost" in this case is referred to the so-called cost function and the quantity that we choose to minimize: the sum of the economic cost of both fuel and battery capacity degradation. As stated earlier, this choice would allow us to compare the two-loss even if they are different practices. From (3.14) we have that the total torque required by the vehicle driving wheels at each time instant is a function of the velocity and acceleration profiles. Thus, following from (3.13), the engine and motor torque T_{eng} and T_{mot}, the mechanical braking torque T_{brk} and the cell current $I(t)$ must be optimally chosen to obtain the required total torque T_{w}. These quantities are chosen as the controls for the problem. This choice is reflected in the chosen control vector (Equation (3.35)) that we will be in the OCP specific formulations.

3.4.1 General Optimal Control Problem

The optimal torque split strategy is obtained through the resolution of a nonlinear OCP. Before to dive into the specific OCP problems formulation, we recap the formulation of a general OCP, expressed by:

$$\begin{cases} \min\limits_{u \in \mathcal{U}} J(x(t), u(t)) & \text{cost function} \\ \text{subject to :} & \\ \dot{x}(t) = f(x(t), u(t), t) & \text{system dynamics} \\ b(x(0), x(T)) = 0 & \text{boundary conditions} \\ c(x(t), u(t)) \geq 0 & \text{path constraints} \end{cases}, \tag{3.33}$$

Where $t \in [0, T]$ is the time, and T is the length of the optimization time horizon. The *cost function* is the function we want to minimize solving the problem. The vector x contains the states that have to respect the *system dynamics*. *Boundary conditions* are the initial and final values of the various states on the horizon. Some of the final states may be free. The *path constraints* instead represents limits on the states that must hold for all the duration of the horizon. They also comprehend limits on the controls. Once each of these four ingredients is defined, the optimal control is formulated.

This section defines the common elements between the online and the offline optimal control. Since they eventually differ in some of the elements above, especially online frameworks, we need to enrich the cost function with elements that consider the receding horizon approach.
For both offline and online optimal control problems, we also set some **path constraints** [36]. The battery power (3.18) is equal to the electric power required by the motor (3.16). The lower and upper limits on the State-Of-Charge (SOC) of the battery are $SOC_{\min} = 15\%$ and $SOC_{\max} = 95\%$ respectively. The cell voltage is constrained between $2.4\,V$ and $4.2\,V$, the electrode surface concentrations must remain between $c^i_{\mathrm{s,min}} \leq c^i_{\mathrm{s}} \leq c^i_{\mathrm{s,max}}$, $i = \mathrm{p}, \mathrm{n}$. Finally, the sum of all the torques (from ICE and motor), according to (3.13), is always equal to the required torque from the inverse longitudinal dynamics (3.14).
Concerning the **boundary conditions**, we must define them differently between offline framework and on-line framework, since one of the main differences is about their horizon: the offline case optimizes along the whole dataset, while the on-line must optimize on a sliding time window, with updating its initial (and final) conditions. The boundary conditions are addressed in the next section since they are specific to each problem. Moving to the **cost function**, it includes two main terms: the final total fuel consumption and the final battery capacity loss. In order to transform this problem into a single-objective problem, we would like to uniform their units: we chose to minimize the final monetary expense due to fuel consumption and battery degradation, expressed in euros [36]. In order to do so, the final fuel consumption (expressed in kg) and battery capacity loss (expressed in percentage of battery loss capacity) are multiplied by the two scaling terms Γ_{fuel} and Γ_{age}, respectively. The choice to minimize the monetary expense is also based on the intuition that an economic motivation would effectively implement such a system by car manufacturers. Therefore, economic benefits can be directly reflected the customers. Therefore, the basic cost function, with the two main terms, can be expressed as:

$$J = \int_0^{t_{\mathrm{f}}} \Gamma_{\mathrm{fuel}}\, \dot{m}_{\mathrm{f}}(x(t), u(t)) + \gamma\, \Gamma_{\mathrm{age}} Q_{\mathrm{loss}}\, (x(t_{\mathrm{f}}), u(t_{\mathrm{f}})). \tag{3.34}$$

we have defined a binary variable $\gamma \in \{1, 0\}$ that multiplies the battery aging term. It is defined to switch between frameworks that considers the battery degradation ($\gamma = 1$) and frameworks that account only for the fuel consumption ($\gamma = 0$). A comparison between these two cases will be carried out in Section 3.5.1.

3.4.2 Off-line optimal control problem

When we move from a general off-line optimization to an on-line applicable optimization, the horizon eventually shortens the amount of information with it. To validate the proposed on-line solution, we need to quantify the maximum performance we can achieve. Thus, we formulate and solve a similar optimal control problem over entire driving cycles using the same speed profile instead of an estimation. We will refer to the solution of this problem as the *offline* solution. The first thing to define are the controls, the vector of the controls includes the torques and the current:

$$\boldsymbol{u}(t) = [T_{\mathrm{eng}}(t), T_{\mathrm{mot}}(t), T_{\mathrm{brk}}(t), I(t)]^{\top}. \tag{3.35}$$

This vector remains the same also for the on-line OCP. For the offline problems the **initial conditions** on the battery SOC is always $SOC(0) = 50\%$. But in this case the **boundary conditions** regards also the final values. Since we span all the driving dataset, final SOC must be constrained within the range $50\% - \mathrm{tol} \leq SOC(T) \leq 50\% + \mathrm{tol}$, with $\mathrm{tol} = 2\%$. This choice is due to the type of hybrid vehicles we consider: mild-hybrid electric vehicles. This type of HEV can charge its battery only through regenerative braking. Therefore to guarantee the possibility to use the battery for fuel-saving during multiple road trips maintaining repeatable performances, the battery must not discharge completely. Therefore, it is a common practice that mild HEV features a *charge sustaining* strategy in order to maintain the battery state of charge around an average value of 50%. In the next section, we will see that our framework is no exception in this aspect. The **cost function**, the **path constraints** and the **system dynamics** no not need any additional change w.r.t. the general problem addressed in the previous

Section 3.4.1. The OCP for the off-line framework is formulated as follows.

$$\begin{cases} \min\limits_{u \in \mathcal{U}} J(x(t), u(t)) \\ \text{subject to :} \\ \\ \text{battery dynamics (3.25)} \\ \\ x(0) = x_0 \\ 48\% \leq SOC(T) \leq 52\% \\ \\ \text{torque split (3.13)} \\ 15\% \leq SOC(t) \leq 95\% \\ c^i_{\mathrm{s,min}} \leq c^i_{\mathrm{s}} \leq c^i_{\mathrm{s,max}} \qquad i = \mathrm{p}, \mathrm{n} \end{cases} . \tag{3.36}$$

3.4.3 Online Optimal Control Problem

The framework proposed in this chapter is based on the OCP applied in a receding horizon fashion. The OCP is therefore solved over a finite future horizon $t \in [t_0, t_0 + T]$, note that $T << t_f$ where t_f is the entire length of the dataset. To be implemented in real-time, the speed profile used in the horizon must be estimated. In the proposed EMS the estimaton of the velocity in the future time window $t \in [t_0, t_0 + T]$ is performed by the Co-Driver velocity prediction algorithm (described in 3.2).

As stated earlier, in a mild HEV the battery can be charge only by regenerative braking. Since the horizon is short, (5 seconds in our case), imposing the final SOC as we have done in the off-line optimization case is not possible. Instead we introduce a SOC-dependent term to the **cost function** (3.34).

$$\begin{aligned} J_{ol} = \int_{t_0}^{t_{\mathrm{h}}} & \Gamma_{\mathrm{fuel}}\, \dot{m}_{\mathrm{f}}(x(t), u(t)) + \gamma\, \Gamma_{\mathrm{age}} Q_{\mathrm{loss}}\,(x(t_{\mathrm{h}}), u(t_{\mathrm{h}}))\, dt \\ & - \rho\, \left(SOC(t_{\mathrm{h}}) - SOC^{\mathrm{trg}}_{\mathrm{f}}\right)^2 \end{aligned} \tag{3.37}$$

The additional term is a quadratic penalty pushing the final state of charge $SOC(T)$ close to the target final state of charge SOC_{trg}. The weight $\rho > 0$ determines how much the charge sustaining term pushes back the SOC to the value of SOC_{trg}. The SOC_{trg} must be tuned to a value that is larger than the desired average value of 50%. The weight $\rho > 0$ determines how much the charge sustaining term pushes back the SOC to the average value of 50%. The values of these two parameters are tuned later on. The MPC is executed every 0.5 seconds, in this case the refresh rate correspond to the step of discretization t_s. The **boundary conditions** are constantly updated at each time step, in particular the **initial conditions**: $x(0) = x(t_0)$. The **path constraints** are the same as we have seen in the general problem.

The final on-line OCP is formulated as follows:

$$\begin{cases} \min\limits_{u \in \mathcal{U}} J_{ol}(x(t), u(t)) \\ \text{subject to :} \\ \\ \text{battery dynamics (3.25)} \\ \\ x(0) = x(t_0) \\ \\ \text{torque split (3.13)} \\ 15\% \leq SOC(t) \leq 95\% \\ c^i_{\mathrm{s,min}} \leq c^i_{\mathrm{s}} \leq c^i_{\mathrm{s,max}} \qquad i = \mathrm{p}, \mathrm{n} \end{cases} . \tag{3.38}$$

3.4.4 OCP collocation and solution

All the formulated OCPs are solved using the *direct approach*, in particular the *direct collocation* method is employed [41]. The first necessary step of this method is the discretization of the problem. Once states and controls are discretized through the time, the direct approach consists in formulating the discretized OCP as a Non-Linear optimization Problem (NLP). We then solve the resulting NLP with *Ipopt*, a state-of-the-art solver based on the interior-point algorithm [122].
The time domain is discretized with a constant time sampling $t_{\mathrm{s}} = \frac{t_{\mathrm{N}} - t_0}{N} = 0.5$. The time grid is defined as:

$$\begin{aligned} \mathbb{G}_N = \{t_0 < t_1 < \ldots < t_j < \ldots < t_N = t_{\mathrm{f}}\} \\ t_{\mathrm{j}} = t_0 + j\, t_{\mathrm{s}}, j = 0, \ldots, N \end{aligned} . \tag{3.39}$$

Where N is the number of grid points. Since t_{s} is constant, it is a function of the horizon initial and final time instants, t_0 and t_{N} respectively. We choose $N = 20$ which leads to $t_s = 0.5$ and 21 values if we consider the initial conditions.
The states and the controls are now defined only on the discretization grid and we need to define their discrete version as following:

$$\begin{array}{rcll} x_j & = & x(t_j) & j = 0, \ldots, N \\ u_j & = & u(t_j) & j = 0, \ldots, N \end{array} \tag{3.40}$$

To perform the discretization of the dynamics, we use the Tustin method, that preserves stability and it is immediate to apply. The difference equation system of the battery dynamics (3.25) now become:

$$\begin{aligned} x_{j+1} = \left(\boldsymbol{I}_{n_x} - \frac{t_{\mathrm{s}}}{2}\boldsymbol{A}\right)^{-1} \left(\boldsymbol{I}_{n_x} + \frac{t_{\mathrm{s}}}{2}\boldsymbol{A}\right) x_j + \left[\left(\boldsymbol{I}_{n_x} - \frac{t_{\mathrm{s}}}{2}\boldsymbol{A}\right)^{-1} \boldsymbol{B}\, t_{\mathrm{s}}\right] u_{j+\frac{1}{2}}, \\ j = 0, \ldots, N-1 \end{aligned} \tag{3.41}$$

where $\boldsymbol{I}_{n_x}$ is the $n_x \times n_x$ identity matrix. The matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are the continous state-space system matrices introduced in (3.25). The controls are $\boldsymbol{u}_{\mathrm{j}+\frac{1}{2}} = \frac{\boldsymbol{u}_{\mathrm{j}} + \boldsymbol{u}_{\mathrm{j}+1}}{2}$ represents the controls evaluated in the grid midpoints. The controls are expressed as piecewise costant functions, this means the value is constant between two grid points on the grid $\mathbb{G}_{\mathrm{N}}$. The cost function (that can be)(3.37) or (3.34) are evaluated on each point of $\mathbb{G}_{\mathrm{N}}$.
Concerning the discretization of the constraint, we choose the **multiple shooting method**, that consists of a discretization of the dynamics constraints. Thus, Equation (3.41) is directly applied as constraint in the NLP. Tha final NLP problem is then:

$$\begin{cases} \min_{x,u} \sum_{j=0}^{N-1} \left(\dfrac{J_{ol}\left(x_{j+1}, u_{j+\frac{1}{2}}\right) - J_{ol}\left(x_j, u_{j+\frac{1}{2}}\right)}{2} \right) t_s \\ \text{subject to:} \\ x_0 = x(t_0) \\ \text{discretized dynamics : (3.41)} \\ \text{Torque split:} \\ \left(u_{1,j} - J_{\text{eng}}\, \dot{\omega}_{\text{eng}}(t_j) + u_{2,j}\gamma_{\text{mot}}\right) \eta_{\text{trn}}^{\text{sign}[a_{\text{x}}(t)]}\, \gamma_{\text{trn}}(t)\, \gamma_{\text{axle}} + u_{3,j} - T_{\text{w}}(t_j) = 0 \\ j = 0, \dots, N \\ 15\% \leq SOC_j \leq 95\% \quad j = 0, \dots, N \\ c^i_{\text{s,min}} \leq c^i_{\text{s,j}} \leq c^i_{\text{s,max}} \quad i = \text{p}, \text{n} \end{cases} \tag{3.42}$$

The Jacobian of the constraints and the gradient of the objective function have to be provided to Ipopt, since interior-point is a gradient based method. The exact Hessian of the Lagrangian can be supplied but it is not mandatory. if not provided, ipopt will approximate them with finite differences. The required analytical derivatives are computed using an algorithmic differentiator called ADiGator [124]. We solved both off-line and on-line OCP on Matlab 2020b on a MacBook Pro with a 2,7 GHz Intel i7 processor and 16 Gb of RAM. In the case of online optimal control, the average computation time of all the MPC steps was 0.157 seconds with a standard deviation of 0.11 seconds. As stated earlier, the MPC algorithm is executed every 0.5 seconds. Thus, the obtained computation time makes the algorithm suitable for real-time implementation and less powerful CPUs. It is important to notice that we did not optimize the code execution, we used a matlab script and the computation time can be improved up to a factor 10 with a compiled language like C++. This would be necessary if one wants to implement the algorithms in a less powerful system (like an ARM architecture).

3.5 Final EMS scheme

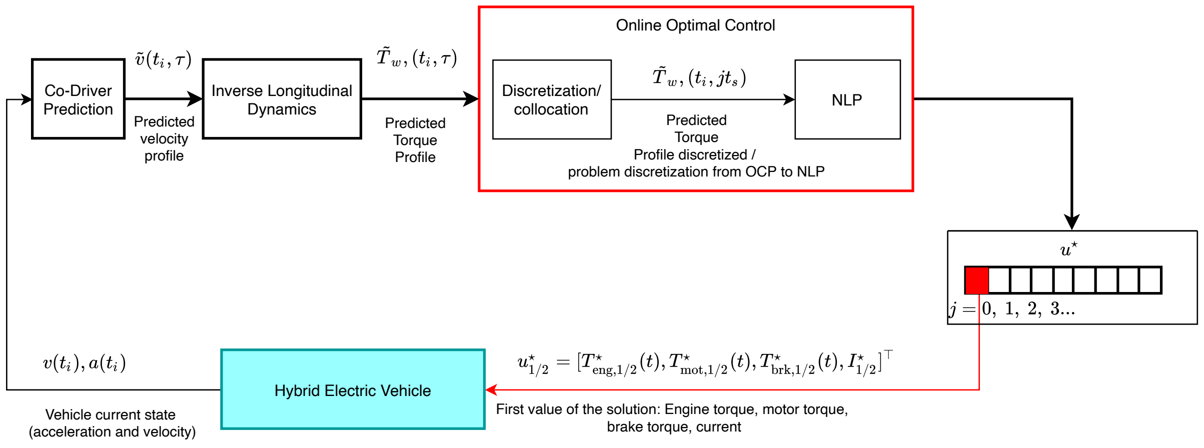
In a canonical MPC formulation, the cost function is minimized w.r.t. the controls a finite time-horizon and only the first value of the optimal solutions are implemented, and then after a time step, the process is repeated. The structure of our EMS follow this pattern, with the additional step of the estimation of the desired outputs in the horizon. In fact, in a generic MPC, the desired trajectory of the outputs is known, but in the EMS case, this is not true. The Co-Driver velocity prediction algorithm (see Section 3.2) is required to estimate the velocity profile that the driver wants to pursue. At each time step (executed every 0.5 seconds), only the first element of the optimal control solution vector (i.e., $u^\star$) is applied to the vehicle powertrain. At the next step, the vehicle state is updated ($x_0 = x(t_0)$), a new speed profile prediction is generated, and the optimization is re-computed for the new window. This MPC scheme allows having a continuously updated knowledge of the system's state thanks to the measurement feedback.
To summarize the overall system structure, in Figure 3.9 a scheme of the EMS is provided.
In Figure 3.9 we can see the overall scheme with the flow of the variables within the algorithm. Given a time step i, at time t_i the ems performs one iteration. At first, the Co-Driver predictor generates the maneuver. They are continuous functions (polynomials) that represents 5 seconds of the estimated future velocity profile $\tilde{v}(t_i, \tau)$ with $\tau \in [0, T]$. This velocity profile is then translated into a Torque request profile $\tilde{T}_w(t_i, \tau)$ using inverse longitudinal dynamics given in (3.14). The torque requested at the wheels is fully given once the velocity and the automatic gear shift policy are static. The OCP problem can now be solved. The torque request profile is sampled at $t_s = 0.5s$ in order to formulate the NLP problem in Equation (3.42). The problem is solved in Matlab using *ipopt* solver. The solution of the problem is expressed as $u^\star$ and it is defined in all the time grid $\mathbb{G}_N$, with $j = 0, \dots, N$. The final phase of the iteration is to apply the solution to the powertrain. Since it is a MPC scheme, only the first element of the solution vector $j = 0$ is applied.

It is essential to notice that, in our scheme, the velocity of the vehicle does not depend on the OCP outputs because the inverse longitudinal dynamics give the total torque and the torque split constraints impose such total torque T_w.

3.5.1 Validation methodology

In this section, we hereby discuss the validation methodology and performance evaluation metrics. The EMS scheme is now validated by simulating the longitudinal vehicle dynamics, the powertrain, and the battery model with the dataset of Section 3.2.1. To evaluate the performances, we decide to compare our EMS with three other frameworks further divided into two sub-framework. The frameworks, including the proposed EMS are associated with a symbol in order to recognize them through the chapter:

- **Co-Driver CD-MPC** This framework is the novel proposed. It uses the Co-Driver predictor 3.2 to estimate the future velocity profile, and it solves the on-line OCP described in Equation (3.38) and discretized in (3.42). Firstly, this framework is simulated using the full cost function in (3.37) $\gamma = 1$. Secondly, the framework is simulated without the aging term, i.e. by setting $\gamma = 0$ in (3.37).
- **Co-Driver 2 seconds CD-MPC-2s** This framework is the same as above, but the horizon length is shortened to 2 seconds in order to investigate how such a difference affects the performances.
- **Exact Prediction MPC 5 seconds (EP-MPC-5s)** This framework shares the same OCP, the same receding horizon scheme, and the same time horizon length with the CD-MPC. The difference is in the knowledge future velocity profile. In EP-MPC, as the name suggests, the desired output of the driver is not predicted, but it is known in advance from the data. The purpose is to evaluate the loss of performances in the EMS due to errors in the velocity prediction. This framework is simulated using both the battery degradation in the cost function included $\gamma = 1$ and not included $\gamma = 0$ in Equation (3.37).
- **Exact Prediction (EP-MPC-10s)** This framework is again based on the exact prediction given by the data, but the horizon is now 10 seconds. This framework is introduced to investigate the importance of horizon length. Also in this case both the options on battery aging term are simulated: $\gamma = 1$ and $\gamma = 0$ in Equation (3.37).
- **Exact Prediction MPC 2 seconds (EP-MPC-2s)** Same as above but shortening the horizon length up to 2 seconds. Again we want to investigate how such a short horizon affects the system's performance.
- **off-line OCP OF-OCP** this framework consists in the off-line problem formalized in Equation (3.36). The offline OCP problem formulated in Section 3.4.2 is solved on the entire driving dataset horizon (6 minutes for each driving record) twice. The framework is simulated with and without the battery degradation cost, i.e., setting $\gamma = 1$ and $\gamma = 0$ in (3.34), respectively. This framework serves as a benchmark of the best



Online Optimal Control
Co-Driver Prediction
$\tilde{v}(t_i, \tau)$
Predicted velocity profile
Inverse Longitudinal Dynamics
$\tilde{T}_w, (t_i, \tau)$
Predicted Torque Profile
Discretization/ collocation
$\tilde{T}_w, (t_i, jt_s)$
Predicted Torque Profile discretized / problem discretization from OCP to NLP
NLP
$u^\star$
$j = 0, 1, 2, 3...$
$v(t_i), a(t_i)$
Vehicle current state (acceleration and velocity)
Hybrid Electric Vehicle
$u^\star_{1/2} = [T^\star_{\text{eng},1/2}(t), T^\star_{\text{mot},1/2}(t), T^\star_{\text{brk},1/2}(t), I^\star_{1/2}]^\top$
First value of the solution: Engine torque, motor torque, brake torque, current

Figure 3.9: Scheme of the EMS, continuous profiles

possible performances achievable, represented by the solution of this framework. Note that the discretization of the (3.36).

The driving datasets used in the simulations were presented in Section 3.2.1.

Charge sustain parameter tuning

Before the validation, we need to tune the necessary parameters in the online frameworks. The parameters to tune the parameters ρ and $SOC_{\mathrm{f}}^{\mathrm{trg}}$ that are in the on-line cost function in (3.37). In the literature, the term "charge sustainability" refers to strategies that aim to keep SOC at a value that allows always to have some energy of the battery available, thus avoiding the battery from continuously or completely discharged during driving. Charge sustaining strategies are not necessarily present in plug-in hybrid vehicles. Such vehicles can be recharged when not used, and therefore the battery capacity can be fully exploited every time the vehicle is used. We want to keep the State Of Charge (SOC) around the initial value of 50%. The charge sustaining strategy is also employed for another reason: to keep the comparison between simulation results consistent. Let us take an example. If one simulation ends with 50% of SOC and another one with 20%, the second simulation would have saved much more fuel since it used some of the battery stored energy without restoring it, such amount of energy would be restored next time the vehicle is used. Therefore, a comparison of the fuel-saving results of the two simulations would not be fair. As stated earlier, we cannot apply the same final state constraints used in the offline framework, and therefore to keep the final SOC close to the initial, we need a charge sustaining strategy.
In all the online frameworks, the parameter $SOC_{\mathrm{f}}^{\mathrm{trg}}$ is set to 60%. The term ρ instead depends on the presence of the battery degradation cost:

$$\begin{cases} \rho = 0.3 & \text{if} \quad \gamma = 1 \\ \rho = 0.8 & \text{if} \quad \gamma = 0 \end{cases} . \tag{3.43}$$

The value of ρ is higher when the degradation is not considered; intuitively, The degradation term partially inhibits the usage of the battery in favor of its lifespan, helping the sustaining strategy itself.

3.5.2 Simulation methodology

We hereby detail how the driving datasets and the vehicle model are employed to simulate the various frameworks. The objective of the simulations is to employ the driving data presented in Section 3.2.1 to drive a vehicle modeled by our model of the dynamics (and powertrain).
We start from the OF-OCP framework, which requires a dedicated discussion. It is closely related to the work in [36], in which the validation phase is based only on standard driving cycles. In this chapter, using the OF-OCP framework, As stated earlier, the results of the OF frameworks are a benchmark for the online frameworks, especially for the proposed one (CD-MPC). Finally, conversely to the MPC-based frameworks, the input of the OF frameworks are the entire velocity and acceleration profiles of each dataset. It returns the solution of the offline OCP (see Section 3.4.2) on the entire driving record (for all the records).
The online (or MPC-based) frameworks perform optimization at every time step. They require the current velocity and acceleration and the velocity prediction (estimated by the predictor or given by the dataset) for the time horizon.
Figure 3.10 represents the overall data flow. This data-flow is valid for all the frameworks. To represent the specific case of OF-OCP, we need to take the horizon length taking $t_i = 0$ and $\tilde{v}(0, \tau) = v(\tau)$ with $\tau = [0, T_{\mathrm{all}}]$, covering all the dataset in one shot of OCP.

In the proposed framework CD-MPC, for each simulation step i at the time instant t_i, current speed $v(t_i)$ and acceleration $a(t_i)$ are sent to the Co-Driver to estimate the future velocity profile $\tilde{v}(t_i, \tau)$. For the on-line frameworks, EP-MPC-5S and EP-MPC-10s, the prediction is exact, and it is taken from the dataset $\tilde{v}(t_i, \tau) = v(t_i + \tau)$, which is the exact prediction. The exact prediction is not continuous, but to keep the notation uniform we keep τ as the variable that spans the time horizon of the predictions.
As stated earlier, we use Equation (3.14) (inverse longitudinal dynamics) to compute the total torque request profile $\tilde{T}_{\mathrm{w}}(t_i, \tau)$ from the velocity and acceleration profiles (exact or estimated). For the receding horizon frameworks xx-MPC, for each instant t_i, one optimization is performed to obtain the optimal controls values. One may notice that the vehicle dynamics is used only to retrieve the required torque. We do not simulate the vehicle motion

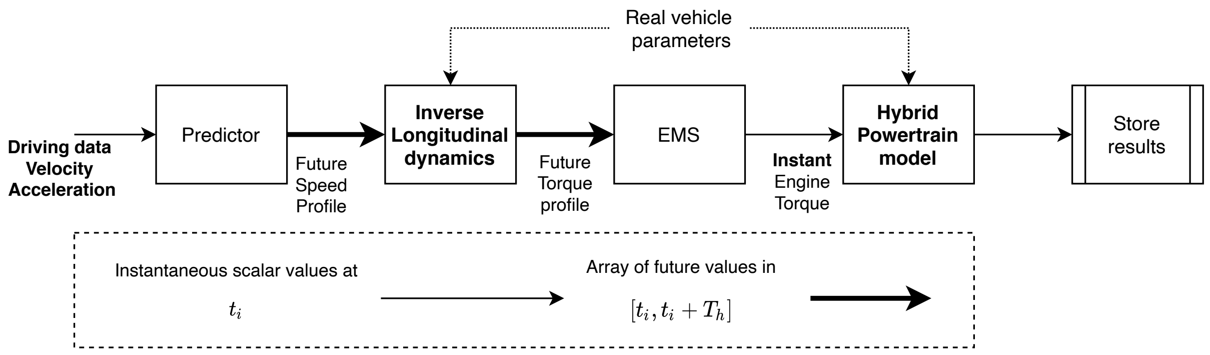


Figure 3.10: Data flow of the simulation

integrating the obtained wheel torque $T_w(t_i)$. Such (instantaneous) obtained torque does not change among the presented frameworks, thanks to the torque split constraints, given by Equation (3.13). The longitudinal kinematics of the vehicle is preserved since the current speed and velocity are equal to the first values provided to the EMS for the torque optimization:

$$\begin{cases} \tilde{v}(t_i,0) & = & v(t_i) \\ \tilde{a}(t_i,0) & = & a(t_i). \end{cases} \tag{3.44}$$

It is implicit the assumption that the internal dynamics of the powertrain do not affect the vehicle motion (except, obviously, for the wheel torque T_w). Conversely to the longitudinal kinematics, the inputs of the power-train depend on the results of the MPC.
The MPC optimally splits the total torque between the ICE, the electric motor and the mechanical brakes along the receding horizon $\tau \in [0, T]$ at each step t_i. The refresh time of the MPC is 0.5s.

3.5.3 Metrics

To evaluate the simulation results, we define some performance metrics. we define some metrics. The first reason to couple an electric motor and a battery with a ICE in a vehicle is to save some fuel. It is possible in two ways: recovering energy from braking and forcing the ICE to work at better conditions, helped by the torque exerted by the electric motor. Therefore, the first metric is the fuel saving percentage, expressed by

$$\mathrm{FSP} = 100 \cdot \left(1 - \frac{\mathrm{m}_{\mathrm{f,HYB}}}{\mathrm{m}_{\mathrm{f,ICE}}}\right), \tag{3.45}$$

where $m_{\mathrm{f,HYB}}$ is the fuel mass in Kg consumed by the hybrid vehicle. The quantity $m_{\mathrm{f,ICE}}$ is the fuel consumption that the same vehicle would need in the case only the internal combustion engine was used. The latter quantity is computed by simulating the longitudinal dynamics of the powertrain on the driving dataset while imposing $T_{\mathrm{mot}}(t) = 0$ in Equation (3.13). The equivalent ICE-only scheme is depicted in Figure 3.11.

The second purpose of the system represents one of the novelties we introduced: the possibility to improve the battery lifespan by minimizing capacity degradation. This is performed featuring a state-of-the-art battery electrochemical model (see Section 3.3.3), which was never used for online optimization. The second metric we

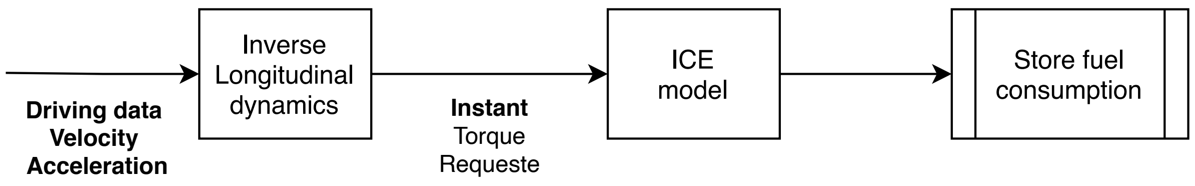

Inverse Longitudinal dynamics
ICE model
Store fuel consumption
Driving data Velocity Acceleration
Instant Torque Requeste

Figure 3.11: Data flow of the computation of $m_{\mathrm{f,ICE}}$.

consider is the relative capacity loss percentage of the battery, which is defined as

$$\mathrm{RCLP} = 100 \cdot \left(1 - \frac{\mathrm{Q}_{\mathrm{f,loss}}^{\gamma=1}}{\mathrm{Q}_{\mathrm{f,loss}}^{\gamma=0}}\right), \tag{3.46}$$

where $Q_{\mathrm{f,loss}}^{\gamma=0}$ is the final capacity loss in the framework cases where capacity loss is not minimized, and $Q_{\mathrm{f,loss}}^{\gamma=1}$ where capacity loss is minimized. The RCLP metric can be seen as the percentage of the battery's capacity that is saved on a driving set by introducing the aging term in the cost function, i.e., minimizing the battery capacity loss. The metrics are collected for all the frameworks: CD-MPC, EP-MPC-5s, EP-MPC-10s and OF. Every framework is simulated with and without the capacity loss minimization ($\gamma = 1$ and $\gamma = 0$ respectively) for a total of 8 frameworks. This metric will be particularly effective in investigating the importance of the battery degradation term in the cost function and its dependency on the horizon knowledge. The two metrics just discussed are dimensionless quantities based on the actual fuel consumption and the actual capacity loss in various cases. These quantities with their units are also reported in the next section to reference the simulation results' actual values. The fuel consumption is reported in Km/l to be immediately readable.
The last metric is the monetary savings per km C_{km}, which is very close to what we used as a cost function. In practice the weight of each term of the cost function is weighted by their price in euro, and this is also evaluated for the ICE to compute the difference:

$$C_{km} = (\Gamma_{\mathrm{fuel}}\, M_{\mathrm{f,km}} + \Gamma_{\mathrm{age}} Q_{\mathrm{loss,km}}) - \Gamma_{\mathrm{fuel}}\, m_{\mathrm{f,ICE,km}} \tag{3.47}$$

Where Γ_{fuel} and Γ_{age} are the cost in euro per unit of fuel and capacity loss respectively. We decide to express the fuel consumption $M_{\mathrm{f,km}}$, and the capacity loss, $Q_{\mathrm{loss,km}}$ per kilometer (on average).

3.6 Results

In this section, we present the results applying the simulation methodology that we introduced in Section 3.5.2 and the metrics discussed in Section 3.5.3.
In Figure 3.12 the fuel efficiency is reported for each driving record and each framework divided by the color. Firstly, we consider the presence of the battery degradation term (setting $\gamma = 1$). The fuel efficiency is reported again for each frameworks removing the aging term shown in Figure 3.13 (setting $\gamma = 0$). These two figures allow us to draw some qualitative considerations. We can notice in both plots that the knowledge of the entire future velocity profile improves fuel efficiency (see OF framework). This relation is emphasized by adding the battery degradation term, i.e., when the optimization minimizes battery degradation. In fact, in Figure 3.12 the MPC frameworks fuel efficiencies (CD-MPC, EP-MPC5s/10s) tend to be farther of the corresponding OF fuel efficiencies with respect to the results without battery degradation term, shown in Figure 3.13. We can barely distinguish between different horizons' results within the same frameworks (all EP-xx-x and all CD-xx-x), showing that the accuracy of the maneuver is more effective than its length for short-horizons. At least for fuel consumption.
Beyond the qualitative considerations made from the fuel consumption values, we can now draw quantitative conclusions looking at more aggregated results. The average Fuel Saving Percentage (FSP) among all the 30

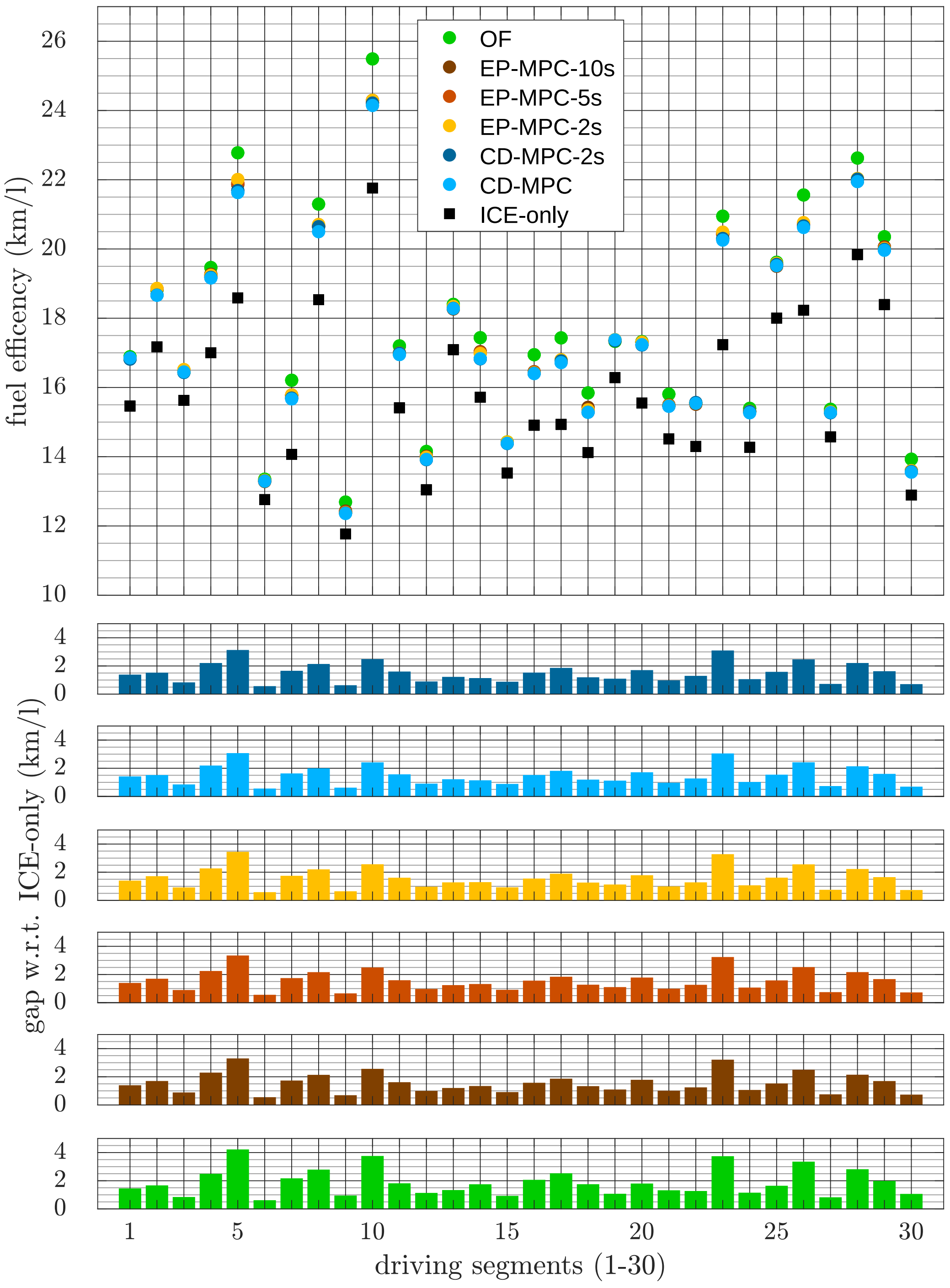


Figure 3.12: Fuel efficiency in km/l for each framework and for each driving record. case $\gamma = 1$ (presence of the degradation term). ICE-only case consumption is reported

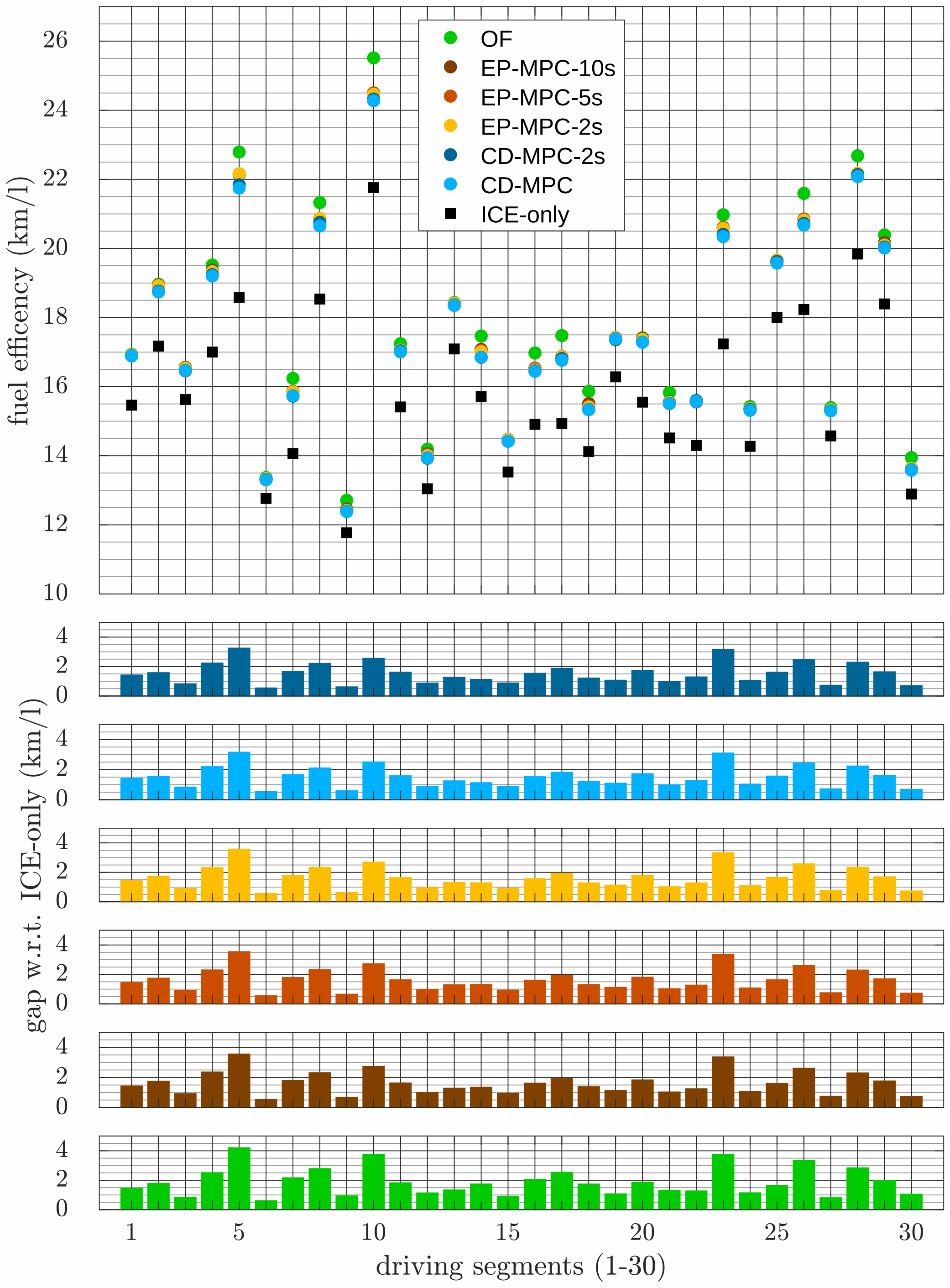


Figure 3.13: Fuel efficiency in km/l for each framework and for each driving segment considering $\gamma = 0$ (without the aging term in the optimization). The ICE-only framework is included as reference.

Battery degradation term	CD-MPC-2s	CD-MPC	EP-MPC-2s	-5s	-10s	OF-OCP
Yes($\gamma = 1$)	8.26%	$\mathbf{8.16\%}$	8.57%	8.50%	8.51%	10.01%
No($\gamma = 0$)	8.53%	8.43%	8.90%	8.92%	8.98%	10.19%

Table 3.5: Average fuel savings with respect to the ICE-only case.

CD-MPC vs:	EP-MPC-5s	EP-MPC-10s	OF-OCP
null hyphosesis rejected	yes	yes	yes
p value	$2.19 \cdot 10^{-6}$	$4.08 \cdot 10^{-6}$	$1.41 \cdot 10^{-7}$

Table 3.6: Results of paired t-test between CD-MPC and the other frameworks Fuel Saving Percentage.

driving records is computed for each of the presented frameworks and the results are shown in Table 3.5, the FSP expression is in Equation (3.45).
The results in terms of fuel-saving are reported in 3.5. As expected, the fuel efficiency decreases as we get farther away from the ideal case provided by the OF-OCP framework.

3.6.1 FSP Statistical tests

We need now to test the difference between the mean values of the obtained FSP. We decide first to evaluate the proposed framework CD-MPC. So the first question to answer is: is the difference between the FSP obtained framework CD-MPC and the one obtained in the other frameworks significant? (these tests do not include the 2 seconds frameworks). This question is answered using the paired student t-test. results are reported in Table 3.6. According to the test results, the means are considered different, Allowing us to discuss the values of Table 3.5. Now we are interested in testing the dfferences between frameworks freaturing different horizons, results are reported in Table 3.7. This test is repeated for both values of γ, p values are not reported for brevity.
The paired t-test between EP-MPC-5S and EP-MPC-10S provides that we cannot reject the null hypothesis that the mean is their distribution present the same mean (p value $= 0.56$). We are confident in using this result to validate the choice of 5 seconds as the proposed horizon. We still propose 5 seconds as a time horizon since the mean is different from 5 to 2 seconds. We can conclude from Table 3.7 that shortening the horizon up to 2 seconds does make a difference. Such differences are discussed in Section 3.6.2. We are also engaging in a comparison between the frameworks with and without the aging term to see if preservation of the battery affects the fuel efficiency. Before to make the comparison, we need to test the difference between the means. The results are shown in Table 3.8, the test rejects the null hypothesis for all the couples of frameworks, the means are therefore different.

3.6.2 FSP considerations

The results are reported in Table 3.5. The highest gap lies between the OF and the EP-MPC-10S, where about 15% of the potential fuel saving is lost. Most of the fuel-saving loss occurs when passing from the knowledge of the velocity profile on the entire horizon (as in OF) to the receding horizon MPC frameworks. Conversely, the fuel-saving performances of EP-MPC-10S and EP-MPC-5S are very similar. We use the latter result to validate the choice of setting the horizon length of the velocity prediction in the CD-MPC to 5 seconds. Indeed, increasing the prediction horizon up to 10 seconds in the CD-MPC framework would not improve the fuel savings since no improvement happens even with the exact predictions of EP-MPC-10S and EP-MPC-5S. The average fuel saving of the CD-MPC framework is slightly lower than the EP-MPC-5S framework (which uses the same horizon length but perfect prediction), showing that the bio-inspired velocity prediction is suitable to feed the online EMS. Finally,

t-test	CD-MPC vs: CD-MPC-2s	EP-MPC-5s vs -2s	EP-MPC-5s vs 10s
$\gamma = 1$	yes	yes	no
$\gamma = 0$	yes	yes	no

Table 3.7: Results of paired t-test between CD-MPC and the other frameworks Fuel Saving Percentage.

$\gamma = 1$ vs $\gamma = 0$	CD-MPC	-2s	EP-MPC-2s	-5s	-10s	OF-OCP
null hyphosesis rejected	yes	yes	yes	yes	yes	yes

Table 3.8: Paired t-test between FSP of the frameworks with aging term and frameworks without

CD-MPC vs:	EP-MPC-5s	EP-MPC-10s	OF-OCP
null hyphosesis rejected	yes	yes	yes

Table 3.9: Results of paired t-test between CD-MPC and the other frameworks for capacity loss.

on average, the proposed CD-MPC framework can achieve 81% of the potential fuel saving of the benchmark OF framework. Moving to the second line of Table 3.5, we can see the average fuel savings obtained when in the frameworks the battery aging is not minimized in the cost function. As expected, the fuel-saving is higher in all four frameworks, but the difference is slight concerning their respective battery-aging-aware cases. We can conclude that minimizing the battery aging does not significantly affect the fuel savings performances. This conclusion was also drawn in [36] for the offline case only (OF), on standard driving cycles: here, we extend such conclusion on naturalistic driving data and also on online MPC frameworks. In other words, we prove that the results of [36] are equally valid in the real-time application.
We did not yet discuss the two frameworks with a short horizon. Looking again at Table 3.5, we can notice that in both Co-Driver and Exact prediction frameworks and $\gamma = 1$ the FSP *improves*. This behavior is not intuitive since usually, we expect that a shortened horizon would worsen the performance of a MPC algorithm. To dive into this result, we need to recall that the overall cost function also includes the battery degradation term, and therefore an improvement of fuel efficiency may raise the overall cost. The second row of the same frameworks indeed presents a decay in the Fuel Saving Percentage from 5 seconds to 2 seconds horizons. This means that if we do not consider the battery degradation shortening, the horizon worsens the FSP, suggesting the above improvement came at the cost of more battery degradation. Now we can move to the battery degradation performances.

3.6.3 The battery capacity loss

The second metric regards the Relative Capacity Loss Percentage (RCLP), defined in Equation (3.46). Before to show and dicuss the main metric RCLP, we hereby define the absolute capacity loss as the percentage of the battery capacity loss w.r.t. the total capacity of the battery ($100 \cdot Q^{\gamma=1}_{\text{f,loss}}$). In Figure 3.14 the absolute capacity loss is shown for each driving record and for each framework when the battery-aging minimisation is active (setting $\gamma = 1$). In the same Figure, we show the difference between the corresponding framework but with no battery-aging minimisation ($100 \cdot Q^{\gamma=0}_{\text{f,loss}} - 100 \cdot Q^{\gamma=1}_{\text{f,loss}}$).

3.6.4 Capacity loss Statistical tests

As we have done for the FSP, we repeat the exact same tests for the battery capacity loss. The first test is the CD-MPC frameworks versus the others, in Table 3.9. All the null hyphotesis are rejected. The second test again is about the horizon length. Same frameworks and different horizons lenghts are tested for both aging minimization active and not. Results are in Table 3.10. This time we reject all the null hypotheses. This means that for the battery degradation minimization, even the difference between 5 and 10 seconds (at least assuming perfect prediction) matters. If we remove the battery minimization strategy, intuitively, we cannot reject the null hypothesis in the differences between 2,5 and 10 seconds in the perfect prediction. We cannot assert that capacity loss changes among these frameworks. This may be intuitive since we have no capacity loss-related term in the cost function. What is interesting is the fact that we have a difference between the CD-MPC frameworks at 5

t-test	CD-MPC vs: CD-MPC-2s	EP-MPC-5s vs -2s	EP-MPC-5s vs 10s
$\gamma = 1$	yes	yes	yes
$\gamma = 0$	yes	no	no

Table 3.10: Results of paired t-test between CD-MPC and the other frameworks capacity loss.

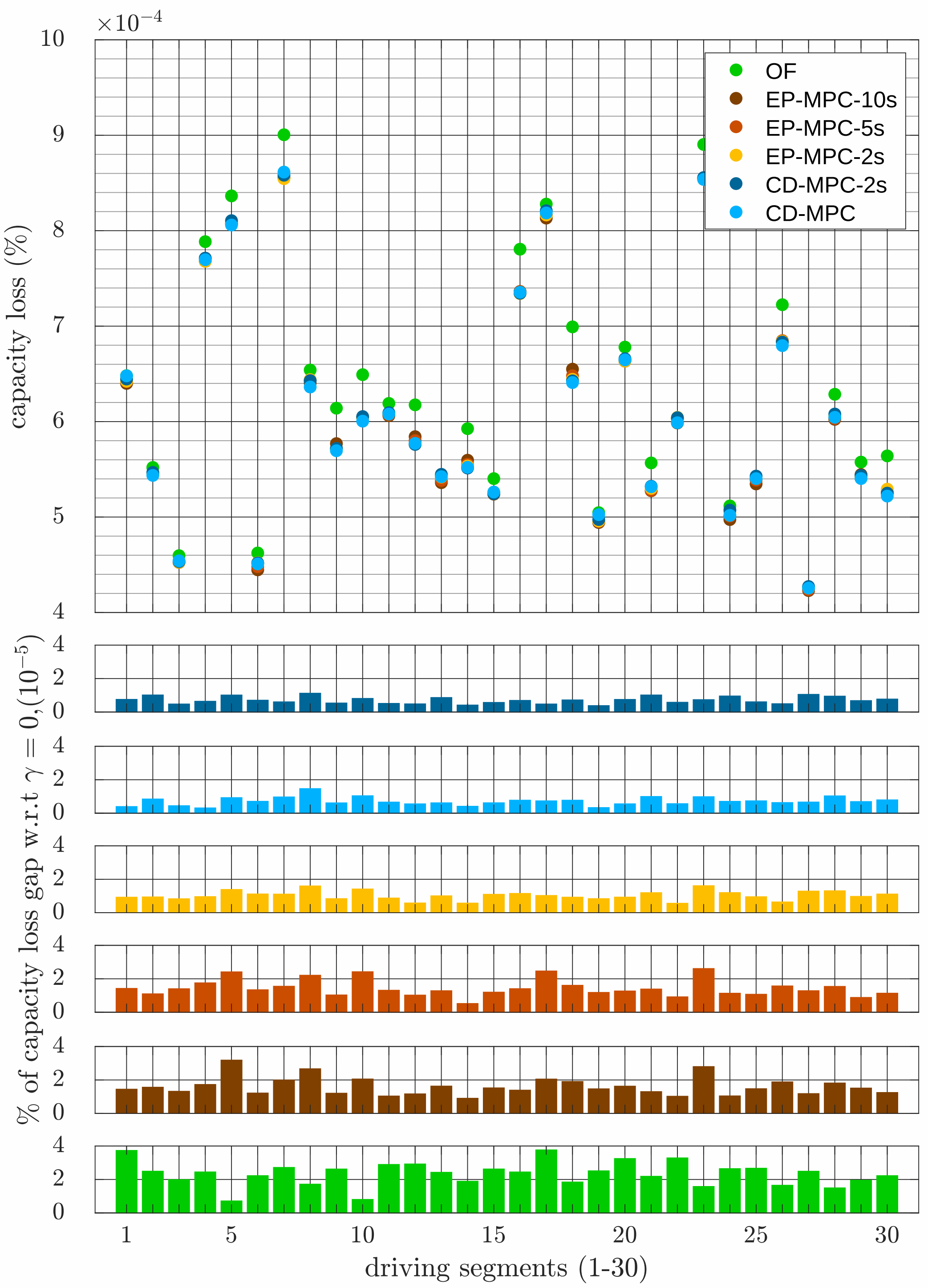


Figure 3.14: Capacity losses for each framework and for each driving record considering ($\gamma = 1$).

RCLP	CD-MPC-2s	CD-MPC	EP-MPC-2s	-5s	-10s	OF-OCP
$(\gamma = 1)$	1.16%	$\mathbf{1.18}\%$	1.70%	2.38%	2.65%	3.71%

Table 3.11: Reduction of capacity loss introducing the aging term in the cost function in each framework. (Relative Capacity Loss Percentage)

RCLP	CD-MPC-2s	CD-MPC	EP-MPC-2s	-5s	-10s	OF-OCP
$(\cdot 10^{-}2$ /km)	$\mathbf{0.032}$	$\mathbf{0.035}$	0.059	0.063	0.065	0.14

Table 3.12: Cost reduction per km for each case (with the aging term), in euro cents.

and 2 seconds. This means that if the prediction is not perfect, shortening the horizon length may affect the bad capacity loss. One may compare Table 3.10 with Table 3.7 and notice the difference in the last column.

3.6.5 Relative Capacity Loss Percentage

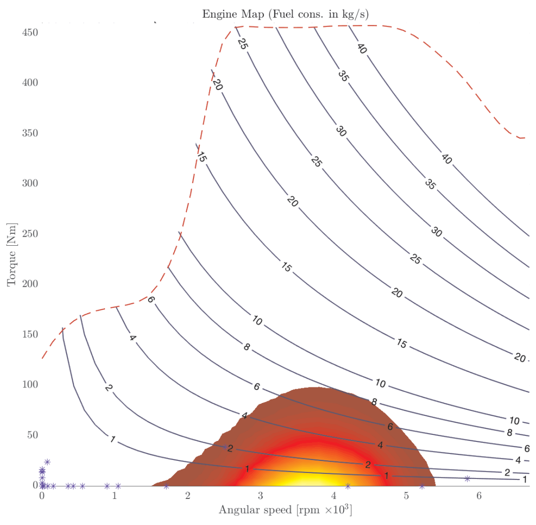
It is difficult to retrieve qualitative considerations from the absolute capacity loss plots. More objective conclusions can be retrieved from the aggregated results of the RCLP metric (see Section 3.5.3). Table 3.11 shows the average RCLP, i.e. the average percentage of the capacity of the battery that is saved introducing the battery degradation term in the cost function (For a more detailed definition of RCLP see Section 3.5.3).
At first, we need to point out that the introduction of the battery degradation term introduces a significant improvement in the battery capacity loss.
As expected, the best results are obtained when all the velocity profile is known, i.e., the case of the OF-OCP framework. From the benchmark framework (OF) to the OF-OCP, only about 30% of the potential capacity saving is achieved. Comparing CD-MPC with its perfect velocity prediction version (EP-MPC-5s), the performance decreases of the 50%. This performance loss is due only to the errors in the velocity prediction since the horizon length is the same.
we can conclude that an accurate speed profile prediction is critical for battery aging minimization and the horizon length.

3.6.6 Monetary savings

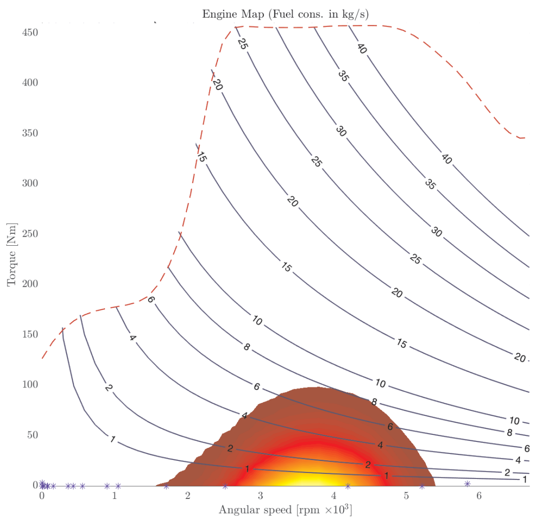
The final metric is the actual monetary cost (in euro) kilometer saved with respect to the ICE-only case. In Table 3.12 the average monetary savings per kilometer are shown. The final cost that is saved is positive but small, both in terms of money and benchmark comparison. The total saved cost is low due to the high cost of the battery, which, even at the optimum, compensates for the fuel-saving. It is worth mentioning that an extension of the battery life may even eliminate the cost of a new battery if its lifespan reaches one of the vehicles. The high difference with the benchmark enhances the considerations made on the capacity loss. On the cost, the offline benchmark benefits of the high capacity recovered with its perfect knowledge of the complete velocity profile.

3.6.7 Engine and motor usage

In Figure 3.15 we show the fuel consumption map of the engine, while the motor efficiency map is drawn in Figure 3.16. On these figures, plots of the engine and motor usage are shown with a color map for the CD-MPC and OF frameworks. The plots of each framework represent the overall usage on all 30 naturalistic driving datasets. Notice that the difference between online (MPC) and offline cases is barely visible. The engine model in this work is quite powerful. Thus it is under-utilized during normal driving. On the other hand, the much less powerful electric motor is fully exploited, especially in high-efficiency regions.

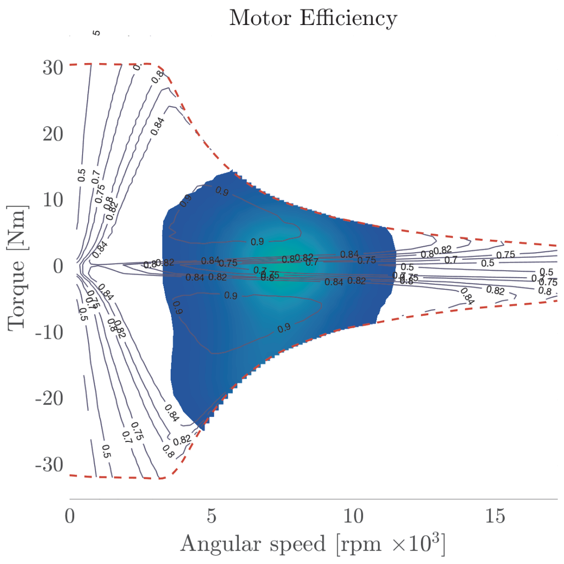


(a)

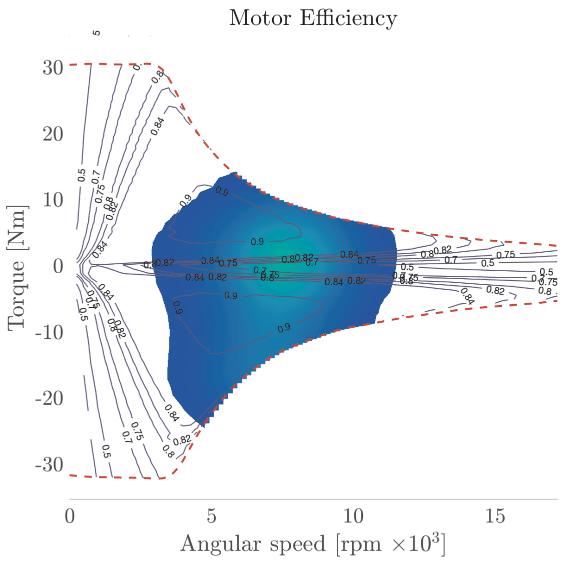


(b)

Figure 3.15: Utilization maps: usage increases from from blue to red. In (a) engine usage in the CD-MPC framework, in (b) the engine usage in the OF framework. All the datasets are considered.



(a)



(b)

Figure 3.16: Utilization maps: usage increases from from blue to red. In (a) motor usage in the CD-MPC framework, in (b) motor usage in the OF framework. All the datasets are considered.

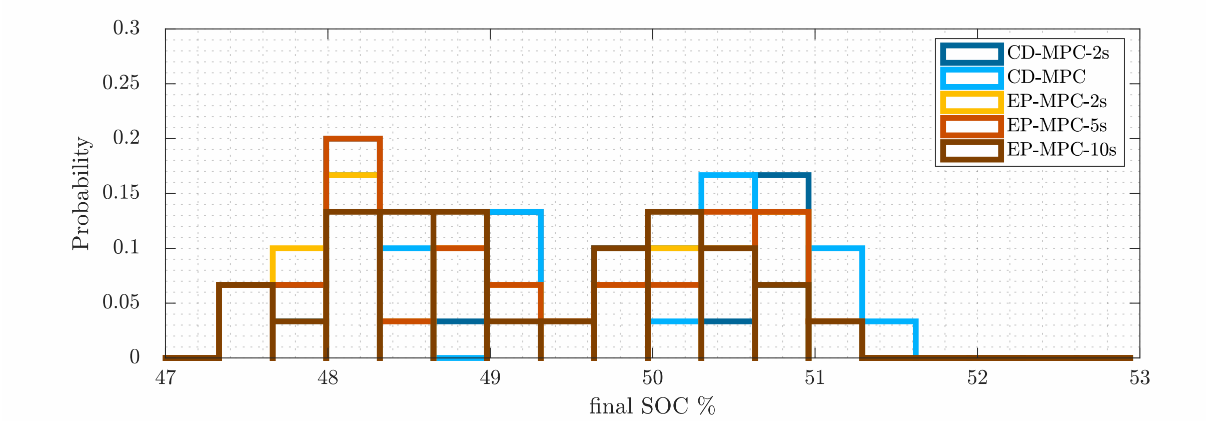


Figure 3.17: Final SOC histograms of the MPC-based frameworks.

3.6.8 SOC related considerations

The SOC evolution in time gives us some information about the difference between the performances of the different frameworks. The first aspect we want to validate is the charge sustain strategy, necessary for the frameworks featuring a receding horizon optimization. Figure 3.17 shows the distribution of the final SOC of the CD-MPC and the other frameworks (considering all the 30 driving records). The final SOC is confined within the window 47%-52%. The charge sustaining strategy is effective since the minimum SOC is bounded above 47%. In Figure 3.18, for all the datasets, the maximum, the minimum, and the median of the SOC at every time instant are plotted for CD-MPC (in blu) and OF-OCP (in orange) frameworks. The two plots in Figure 3.18 presents two differences: the general trend of the median and the difference between the minimum and the maximum evolving in time. Thanks to the knowledge of the entire driving record, the OF-OCP framework can exploit a wider range of the SOC without affecting the final SOC. The right area (orange) in Figure 3.18 ranges down to 40% while the one on the left (blue) does not cross under the value of 47% by the necessary charge sustaining strategy. The impossibility for the CD-MPC framework to exploit lower values of the SOC has a double effect on the battery. Firstly, higher SOC values correspond to higher battery aging (see Equation (3.32)), resulting in a loss of capacity-saving performance between offline and online frameworks. Secondly, the CD-MPC framework frequently cannot fully recover energy from deceleration, resulting in a loss of performance in terms of fuel-saving.

3.7 Conclusions

This paper presented a complete energy management system for hybrid electric vehicles. Firstly, we designed a prediction scheme to estimate 5 seconds of the future velocity profile the driver intends to pursue. Then, we presented an online energy management system to optimally split the requested driving torque between the electric motor and the internal combustion engine. The EMS is based on a state-of-the-art battery model. It considers the battery aging in the form of capacity loss, and it is implemented using optimal control solved through the direct approach. We validated the prediction scheme and then the entire online EMS framework (BP-MPC) on 30 naturalistic driving segments of 6 minutes each from 10 different drivers. In addition, the prediction scheme was also validated on standard driving cycles and compared against two benchmark prediction methods widely employed in the literature. The proposed EMS system was then compared with three ideal benchmarks (one "global" solution (OF) and two MPC-scheme fed by exact velocity predictions (EP-MPC-5S, EP-MPC-10S). Comparison against the benchmarks led to desirable results in reducing fuel consumption. For example, the proposed BP-MPC framework could save 80% of the fuel saved by the ideal offline OF framework. The battery capacity savings resulted in being more sensitive to the accuracy of the velocity prediction. Our novel BP-MPC framework was able to save only the 31% of the potential battery capacity saved by the ideal OF framework and the 50% of the one obtained with the exact prediction EP-MPC-5S framework. Thus, as one of the significant outcomes of this work, it turned out that accurate speed prediction is more relevant for battery aging than for fuel consumption. Moreover, it was shown that the proposed EMS system is real-time feasible, which means it can be

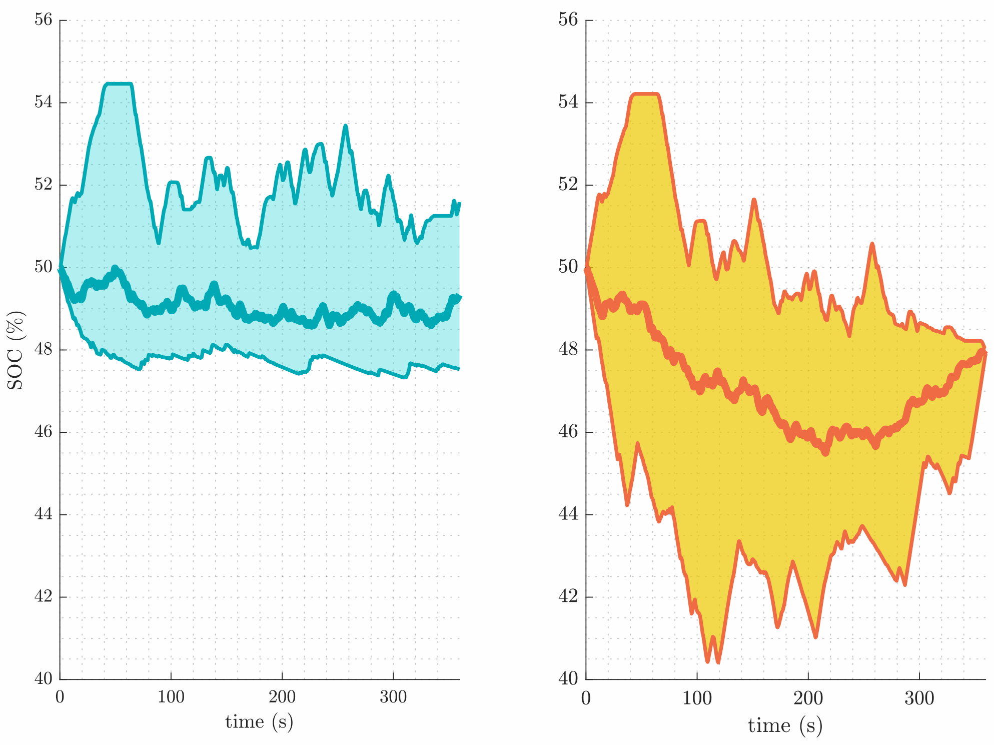


Figure 3.18: SOC evolution in time, the blue left curve is CD-MPC framework, the right orange is OF-OCP framework.

embedded on a vehicle for a practical application.
Despite the latter result, the real-time system extended the battery life, making it suitable for implementation. It is worth noticing that the small amount of money saved is mainly due to the high cost of the battery that has a double effect on the final cost: increasing the overall cost per kilometer and moving the optimal solutions to higher usage of the fuel. Increasing the battery life up to the life of the vehicle itself may significantly reduce the cost for the final user and reduce the battery's environmental impact.
The approach can also be modified by introducing different costs in the objective function, such as an equivalent environmental impact of fuel and battery capacity loss. Although we did not address the problem in this thesis, it is worth mentioning that future development can be the introduction of new inputs to the velocity prediction algorithm, like information from the ADAS of the vehicle. The latter includes the velocity of the preceding vehicle, the presence of intersection and roundabouts, and the readings of the speed limits. To include these features, the predictor would need an ahead road scenario that is correctly perceived with sensors. We can also conclude that increasing the horizon length makes a difference only with a reasonable accuracy since, in the CD-MPC framework, the difference of performance between 2 and 5 seconds was smaller than expected.

Chapter 4

Intersection Management for Connected Autonomous Vehicles

In this chapter, we move from driving assistance systems to automated vehicles. The connected vehicles paradigm is maintained, We hereby exploit the connection between vehicles to address the problem of intersection management for Connected Autonomous Vehicles (CAVs). We propose a framework that maximizes the traffic throughput of a broad category of possible intersections, including multiple lanes and multiple turns options. The objective of the framework is to optimize the time that vehicles require to clear a given arbitrary intersection, taking into account all the vehicles' states and conflicts. The overall problem is formulated as a bi-level optimization problem. In the low level we solve a simple Optimal Control Problem (OCP) for each vehicle, in the high level we formulate a Mixed Integer Non-Linear Programming Problem (MINLP) to retrieve the optimal scheduling and the arrival time of each vehicle. The MINLP problem is then approximated and solved by means of a piecewise linearization technique. The final proposed problem to solve is then a Mixed Integer Linear Programming problem (MINLP). The algorithm is tested on a benchmark scenario, and compared with a typical heuristic strategy.

4.1 Introduction

In the last decade, the concept of automation in the automotive field gained importance. Alongside, communication paradigms such as vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) are rapidly evolving. The confluence of vehicle automation and vehicle communication gave birth to the concept of Connected Autonomous Vehicles (CAV), which can improve the capabilities of self-driving vehicles substantially.
Road intersections play a crucial role in mobility networks, especially in urban areas they affect the safety and efficiency of transportation. The introduction of CAVs opens new ways to address the problem of intersection management. Optimal coordination of CAVs at intersections can improve safety and traffic efficiency. In the last decade, research about intersection optimization gained much interest, with a significant boost in the last three years. Two recent surveys are [95] or [77]. They both present a detailed review of the literature on intersection management systems for autonomous vehicles. Many algorithms address such problems. They differ in several aspects. For instance, the approaches can be divided by the goal of their optimizations. In the literature *safety, efficiency, ecology* and *passenger comfort* are the most widespread goals. Another interesting difference that divides the algorithms into two categories is the communication paradigm. We can find *distributed* approaches and *centralized* approaches. In the centralized case, the management of the intersection must be demanded to some managing unit located in the vicinity of the intersection, where the computations took place. In the distributed case, instead, CAVs try to optimize their goal only by communicating with each other and performing their own

computations without a central unit knowing all the states. A examples of distributed approaches can be found in [25], [84] and [8]. In the first reference, the problem is represented as a generalized Nash equilibrium problem that results from a coupled optimal control problem. Moreover, to the best of our knowledge, this is the first work on intersection scheduling with proof of convergence. In the underlying work, we consider a centralized approach. If no traffic priorities are given, the intersection management problem, in this case, can be addressed as a bi-level optimization problem. On the lower level, the motion of each individual vehicle has to be found in accordance with some optimization goal, whereas on the higher level, one tries to find an optimal crossing order such that no collisions occur, and some global cost is optimized. Often, this global cost is simply the total time elapsed to clear an intersection for all vehicles in consideration. Let us first mention some existing works with similar modeling/solution techniques and then discuss the contribution of our work. The approaches given in [47], [88] exploit the same vehicle model and the same modeling techniques for scheduling constraint. The geometry of the intersection is instead restricted to a specific use case with no turns, and due to some simplifications, the higher level problem is modeled as a MILP. In [67], [68] a more sophisticated car model is considered which results in a MINLP on the higher level. The emerging MINLP is then solved using an SQP-type algorithm. See also [66] for a real-world experiment. In this paper we consider a simple car model as in [47], [88], however, on the higher level we end up with a MINLP as in [67], [68], and in contrast to Hult et al., we solve the MINLP with some linearization technique inspired by [125]. Such a choice is explained by the existence of high-performance software packages for solving MILP. We show that our technique can be used for real-time applications. The rest of the chapter is organized as follows. In Section 4.2 we define the details on the problem we address, discussing restrictions and assumptions we make regarding vehicles and intersection geometry as well as the vehicle model. Sections 4.4 and 4.5 are devoted to the actual problem formulation divided in two levels, low and high respectively. We introduce the vehicle model, define scheduling constraints, and state the MINLP. In Section 4.6 the piecewise linearization technique is introduced, and a final MILP is stated. Finally, in Sections 4.7 and 4.8.1 we present some numerical experiments and state conclusions.

4.2 Problem modeling

4.2.1 Intersection geometry

The approach used in this work is conceived to be flexible and can be easily adapted to different types of intersections. We define some necessary quantities. If these objects can be identified for an intersection, such an intersection is suitable with the proposed approach. In Figure 4.2 the objects are illustrated in a visual example.

Lanes

The lanes that enter the intersections are called entering lanes. The others are exit lanes. In a single lane a vehicle is not allowed to overtake, since we are in the vicinity of the intersection, and the vehicles cannot change lane. We assume they are already on the desired lane.

Paths

Every entering lane is connected to all the exit lanes with predefined paths (see Section 4.2.2). Every vehicle i travels on the path $P(i)$. Their geometry is known. These paths resemble the concept of paths we have seen Section 2.5.

Conflict zone

The Conflict Zone (CZ) is an intersection area that contains all the overlapping points of the paths. All the possible collisions (except rear collisions on the lanes before the intersection) can only occur in this area. This area is the part of the intersection shared between vehicles at different times. The time at which vehicle i reaches this area is defined as t_i.

Conflict matrix

The geometry of the intersections contains paths that can overlap with other paths. The management of these conflicts is the core of the intersection management problem. If there are no conflicts, the global minimum time solutions would be only the sum of the single minimum time trajectories of the vehicles.
The conflict between the paths is of four different types, depending on the entering and exit lanes. This characteristic of pairs of paths is encoded in an MxM matrix called conflict matrix M_c. The entries of the conflict matrix can assume five values (4 for conflicts plus one for no conflict). Given a path k and a path l, the entry k, l of the matrix can assume the following values:

$$M_c(k,l) = \begin{cases} 0 & \textit{paths do not overlap} \\ 1 & \textit{paths conflicts} \quad \textit{different entering lane} \quad \textit{different exit lane} \\ 2 & \textit{paths conflicts} \quad \textit{different entering lane} \quad \textit{same exit lane} \\ 3 & \textit{paths conflicts} \quad \textit{same entering lane} \quad \textit{different exit lane} \\ 4 & \textit{same path} \quad \textit{same entering lane} \quad \textit{same exit lane} \end{cases} \tag{4.1}$$

This matrix encodes the necessary information on which paths are in conflicts. Note that the distinction between the conflict types is necessary to exploit the entire conflict zone to host more than one vehicle simultaneously. A similar example is provided in [117] using a scheme with only one type of constraint, in such case the conflict zone can be occupied only by one vehicle at a time.

Lengths

The geometry of the intersection is closely related to the formulation of the constraints. We hereby define some crucial quantities required to model the conflicts between paths inside the conflict zone. The various length that we are going to define are shown in Figure 4.1. The first length to define is L_k (or L_p) which is simply the length of the path k inside the conflict zone CZ. This length does not depend on the size of the vehicles. More specific lengths need to be defined to model specific conflicts between two paths and two vehicles.
The quantity $L_{a,i,j}$ is defined as the length of the path (inside the conflict zone) that the vehicle i can travel without being overlapped with the path of the vehicle j. Note that this length depends only on the size and path of vehicle i (the vehicle that passes first) and the path chosen by the vehicle j. Instead the length $L_{b,i,j}$ is the length of the path i before the overlapping region of j. It means the path such that vehicle i reaches the overlapping region between path $p(i)$ and $p(j)$ after $L_{b,i,j}T_i$ (assuming constant velocity, Assumption 4.3). In Figure 4.1 examples of these quantity are shown.
Note that these lengths depend on the size of the footprint of vehicles, especially on the length of the vehicles (the width is actually under a certain standard threshold). For the sake of simplicity, we decided to introduce the assumption on the same footprint 4.5. Extension on variable size can be a matter of further work extensions.

4.2.2 Assumptions

Assumption 4.1. CAVs states known *In the problem addressed, N connected autonomous vehicles are considered. They are indicated by the index $i = 1..N$. The states of those vehicles are assumed known without uncertain (The model, with its states, is detailed in Section 4.3).*

Assumption 4.2. Predefined paths *We assume that the vehicles travel on predefined paths. Every entering lane is connected to an exiting lane by a predefined path. Vehicles are allowed only to move along the assigned paths. Paths are enumerated one by one in the intersection, and they are indicated by the letter P and the index $k = 1..M$. The vehicle's path is then known in advance, and it is indicated by the map $P(i) : i \rightarrow k$, which relates the index of the vehicle i to its chosen path k. No lane change is allowed (the vehicle is already in the correct lane) in the portion of the intersection considered by the problem. This assumption is widely used for this problem [95], in*

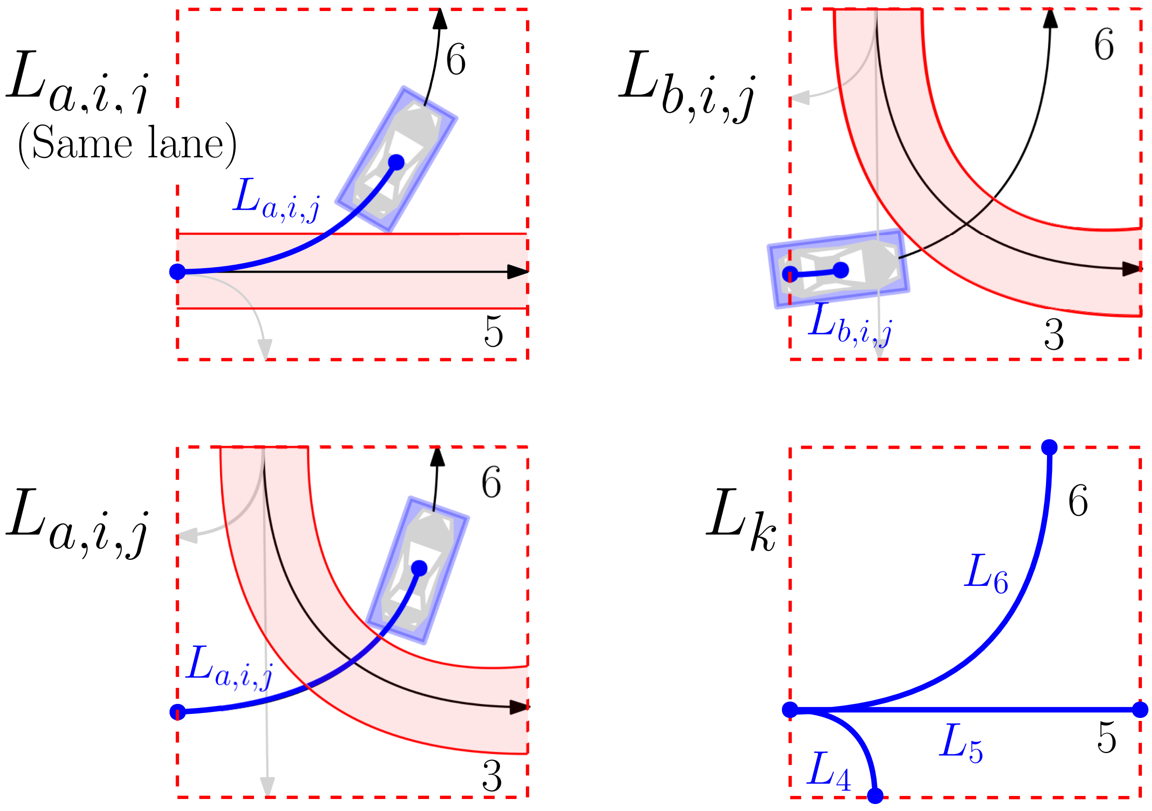


Figure 4.1: Lengths involved in the geometry: the two representations of the CZ on the left contains $L_{a,i,j}$, the length the vehicle i travels before clearing the span of the vehicle j (the latter in red). In the upper right CZ, $L_{b,i,j}$ is the length the vehicle i can travel before the overlapping. General length L_k are provided in the last square for three paths.

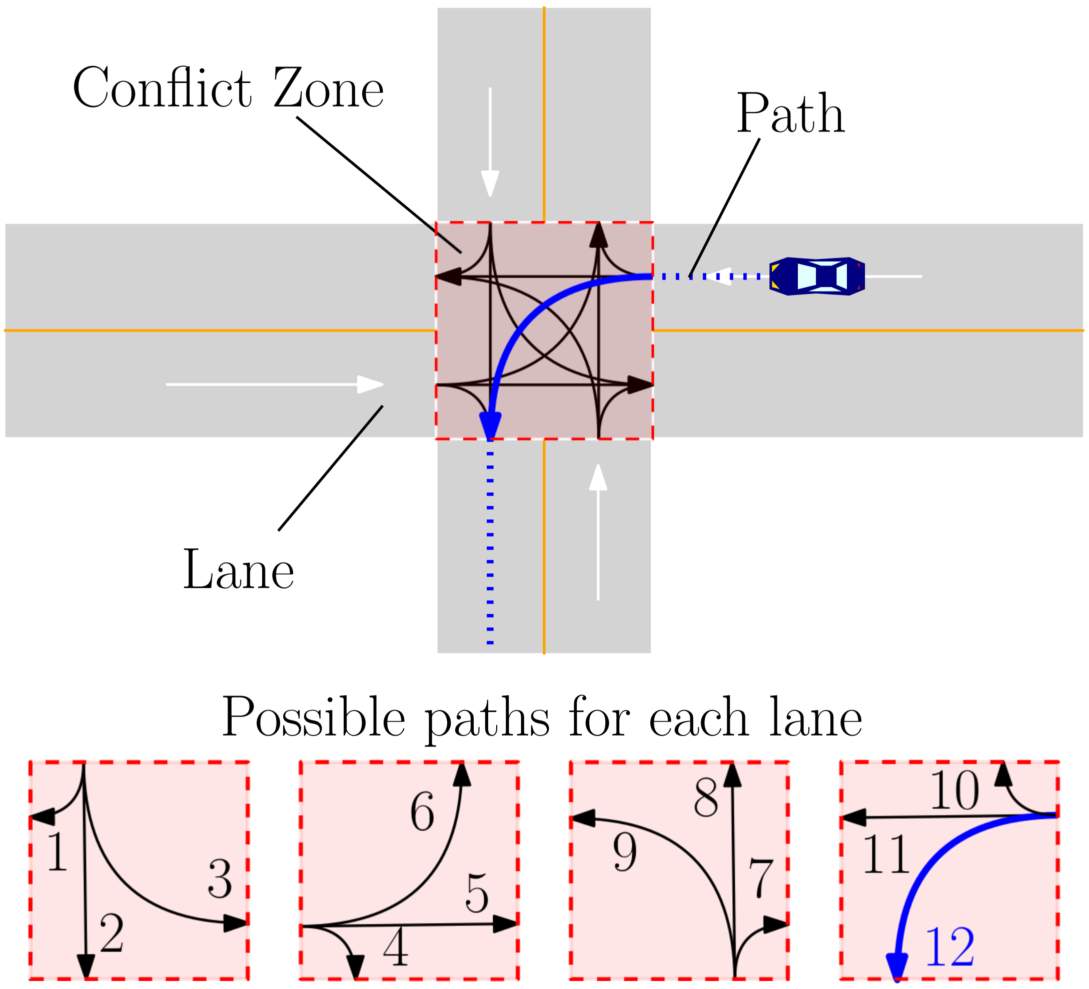


Figure 4.2: Sample of an intersection geometry: in the top of the figure a 4 roads intersection is shown, there are 4 entering lanes and 4 exiting lanes and 12 possible paths. The paths are shown in a detail of the conflict zone in the bottom of the figure. The path of the sample vehicle 1 is the number 12, then $P(1) = 12$

some countries (like Germany) these paths are even painted on the roads as they were lanes. These paths resemble the concept of paths we have used in Section 2.5.

Assumption 4.3. Constant velocity in the conflict zone *Once the conflict zone is defined. The longitudinal motion of the vehicle is divided into three phases. The approaching phase is fully described in 4.3 and regards the motion from the initial position of the vehicle to the conflict zone. The second phase starts in the conflict zone, and the vehicle velocity is assumed to be constant inside this zone. This choice is motivated by the simplicity of the timings inside the conflict zone and the fact that the turning paths limit the velocity. Constant velocity simplifies its limitations of it based on the curvature. This assumption is also widely used in literature [95]. The third phase, after the conflict zone, is not discussed here. The velocity in the conflict zone of the vehicle i is $v_{f,i} = 1/T_i$, being T_i the reciprocal of that velocity (its meaning is detailed in Section 4.4).*

Assumption 4.4. Feasible initial conditions *This assumption is straightforward. The formulated MILP problem cannot be solved for any initial conditions. We exclude initial conditions that lie to an infeasible problem (impossibility to respect all the constraints). This assumption does not imply any loss of generality since these initial conditions eventually lie to an accident, and they must be discarded a priori.*

Assumption 4.5. Same footprint of vehicles *The footprint of each vehicle is different, but we consider a footprint that can fit most of the vehicles. This assumption does not affect the flexibility of the approach, a relatively large footprint can indeed be used in a real application introducing a safety margin.*

4.3 Vehicle model and motion

In this section, we describe the simplified model used for the vehicle. We also explain motion planning, which is divided into three phases.

4.3.1 Vehicle model

For simplicity, the selected vehicle model is a double integrator. The states are the curvilinear position along the predefined path $s(t)$ and its derivative (the longitudinal velocity) $v(t)$. The input of the model is the longitudinal acceleration $a(t)$. The latter is bounded by a_m and a_M as shown in equation (4.2). The zero of the curvilinear coordinate $s(t)$ is the point of the path in which the conflict zone begins. We will see that placing the origin at such a point simplifies the motion planning. In Equation (4.2) the vehicle model is illustrated:

$$\begin{array}{rcl} \dot{s}(t) & = & v(t) \\ \dot{v}(t) & = & a(t) \\ & & \\ a_m \leq & a(t) & \leq a_M \\ 0 \leq & v(t) & \leq v_M \end{array} . \tag{4.2}$$

The velocity is nonnegative and bounded in the model, other limits in velocity, depending on the curvature of the path and on the legal limit, are introduced in the problem formulation. Initial conditions are given (see Equation (4.3)), they are simply the state of the vehicle when the optimization starts ($t = 0$). Note that in (4.3) s_0 needs a change of sign since it is the distance from the conflict zone, and therefore positive.

$$\begin{array}{rcl} s(0) & = & -s_0 \\ v(0) & = & v_0 \end{array} \tag{4.3}$$

There are N vehicles in the intersection, and the state (and initial condition) of each of them is indicated by a pedix i or j. In terms of geometrical characteristics, a vehicle's footprint is represented as a rectangle. The rectangle is conceived to contain the actual dimensions of the vehicle projected on the ground. The position $s(t)$ is the position of the geometrical center of the rectangle. The quantity $s_{0,i}$ is the curvilinear distance (positive) to the conflict zone of the vehicle i at $t = 0$. It needs a minus sign to be a coherent initial condition (The condition $s(t) = 0$ means the vehicle reached the starting point of the conflict zone, and it happens at $t = t_i$).

4.3.2 Motion planning and problem formulation

The problem formulation proposed and solved in this work aims to find the optimal times and the optimal order for the vehicles to arrive at the intersection (t_i) minimizing the overall time elapsed (that is equivalent to maximize the traffic throughput). The longitudinal motion of the vehicle along the curvilinear coordinate $s(t)$ is divided into 3 phases:

- Approaching phase, $t \in [0, t_i]$
- Crossing phase, $t \in \big(t_i, t_i + L_{(p(i))}T_i\big)$
- Leave intersection $t > T_i$

The optimization concerns the first two phases. In the approaching phase, the vehicle's motion is computed by a low level optimal control problem in a parametric fashion. The duration of this phase is t_i, which the high level planner gives.
In the CZ instead, we have the crossing phase, where the vehicles maintain constant velocity ($v_{f,i} = 1/T_i$), such velocity is also the "arrival" velocity $v_{f,i} = v(t_i)$. The third phase is not considered since the vehicle can continue its travel without affecting the intersection. What we are going to use is the *reciprocal* of the velocity:

$$T_i = \frac{1}{v_{f,i}}. \tag{4.4}$$

The parameter T_i is defined as the reciprocal of the arrival velocity of the vehicle i. It can be seen as the time required to travel 1 meter inside the conflict zone (the velocity in the conflict zone is constant see 4.3). The time $t_i + L_{(p(i))}T_i$ is the sum of the approaching and crossing phases times and it is how long the vehicle i takes to clear the intersection. The approaching and the crossing phase durations sum up to the total time required by the vehicle i to clear the intersection:

$$t_i + L_k/v_i(t_i) = t_i + L_{P(i)}T_i. \tag{4.5}$$

Where L_k is the length of the path $k = P(i)$ in the Conflict Zone. Since t_i is given from the scheduling, the low level problem provides the final velocity $v_i(t_i)$.
The selected longitudinal motion is the solution of an optimal control problem (OCP) that maximizes the final velocity (and therefore minimizes the overall time given by (4.5)). The OCP is solved analytically using the Pontryagin Minimum Principle.
Equation (4.6) is parametric in t_i, and it can be used directly in the scheduling problem through the reciprocal of the final velocity $T_i(t_i)$.
In short, the motion planning of the low level gives the non-linear function $T_i(t_i)$ for the scheduling problem of the high level. Since T_i is the reciprocal of the final velocity of the planned motion.
This function $T_i(t_i)$ plays a central role in linking the optimal control level to the scheduling level, delegating the global optimization to the high level. In Figure 4.3 this bi-level scheme is depicted.
Additional limits on velocity The function $T_i(t_i)$ can be further modified by the constraints on velocity. This is possible because the motion planning can be modified for final velocities lower than the maximum possible. In particular, additional limits can be used to avoid unfeasible velocities. In Section 4.4.3

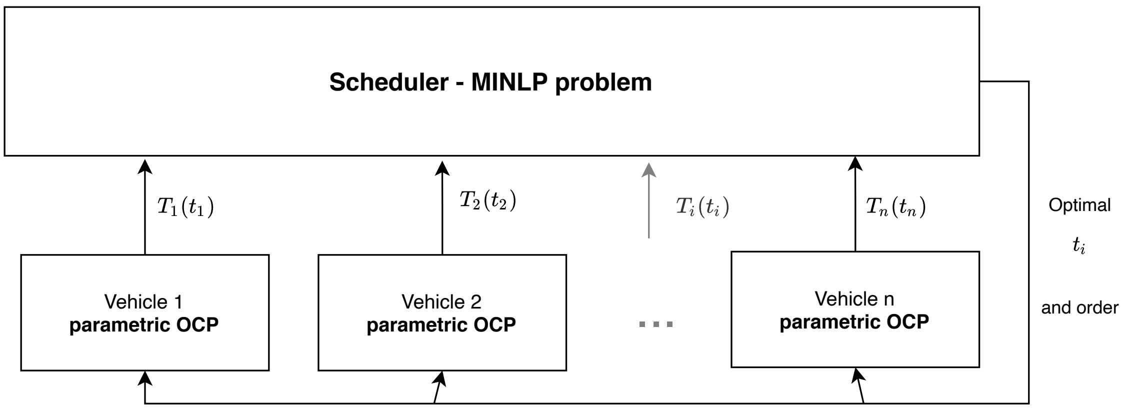


Figure 4.3: Scheme of the approach, the high level optimization variables are t_i and the order of the vehicles (see Section 4.5), while the low level consists in parametric solutions of OCP problems that provides $T_i(t_i)$ (See Section 4.4). The cost function of the high level's optimization problem contains $T_i(t_i)$.

Symbol	Description	Section
$i, j \in \mathcal{V}$	Vehicle and index set	4.3
N	Number of vehicles	4.3
$k, l \in \mathcal{P}$	Path indices and index set	4.2
M	Number of paths	4.2
$P(i)$	Path chosen by to vehicle i	4.2
$M_{C\ k,l}$	k, l conflict matrix entry	4.2
$s_i(t)$	Position of vehicle i along its path	4.3
$v_i(t)$	Velocity of vehicle i	4.3
$a_i(t)$	Acceleration (control variable) of vehicle i	4.3
$a_{m,i}; a_{M,i}$	Acceleration bounds of vehicle i	4.3
t_i	Arrival time of vehicle i	4.3
$t_{i,min}$ $t_{i,max}$	Lower bound of t_i Lower and upper bounds of t_i	4.3
$v_i(\cdot)$	Arrival velocity of vehicle i	4.3
$v_{P(i)}$	Maximum allowed velocity on path $P(i)$	4.3
$I_{t_a}^{t_n}$	Time interval $[t_a, t_b]$	4.3
$T_i(\cdot)$	Inverse of the arrival velocity of vehicle i	4.3
$t_{i,\star}$	Velocity saturation starting time of vehicle i	4.3
C_{large}	Some very large constant	4.5.1
w_{ij}	Binary decision variable (equals to 1 if vehicle i goes first)	4.5.1
L_{ij}^b	Length to travel before overlapping zone of $P(i)$ and $P(j)$	4.5.1
L_{ij}^a	Length to travel to clear overlapping zone of $P(i)$ and $P(j)$	4.5.1
L_k	Length of path k in CZ	4.5.1
$T_i^{headway}(\cdot)$	Time to reach maximum velocity in exit lane for vehicle i	4.5.1
$T^{gap}(\cdot)$	general time gap to avoid real collision in the approaching phase i	4.5.1
v_{exit}	Maximum allowed velocity in exit lanes	4.5.1
$T_i^{switch}(\cdot)$	Switching time point from deceleration to acceleration	4.5.1
$\mathcal{T}_i$	Time interval discretization corresponding to vehicle i	4.6
K_i	Number of discretization points of i-th interval	4.6
$z_{i,k}$	Pointer variable corresponding to i-th interval	4.6
$\lambda_{i,k}$	Lambda variable corresponding to i-th interval	4.6
$T_{i,pw}, v_{i,pw}$	Linear approximations of $T_i(\cdot)$ and $v_i(\cdot)$	4.6

Table 4.1: Quantities and variables introduced in this chapter

4.4 low level problem

The intersection management problem is hereby treated as a bi-level optimization problem. The lower level of this problem regards the motion of the single vehicle. In the formulation of the high level problem, we decided to put as variables the time to arrive at the intersection of the vehicle i, (t_i). The velocity in the intersection is constant and it is equal to $v_{f,i} = 1/T_i$, note that T_i represents *the time for the vehicle i to travel one meter in the conflict zone*.
The constraints on the value of t_i are defined in Section 4.5, and they do not depend on the type of motion, since the minimum t_i, defined as t_{start}, is given by a constant maximum acceleration motion.
Instead, the approaching phase can be traveled by any type of motion (according to the limit of the vehicle). In our problem, we decided to pursue the minimum time to clear the intersection as a first approach to the problem, and we did not take into account the comfort of the passengers. It is important to recall that the proposed approach can be applied with motion laws other than the one proposed in this work.

4.4.1 Maximize final velocity OCP

The first thing to take into account is that the time to arrive t_i is an argument of the high level problem (ref). Therefore we cannot actually minimize it in the approaching phase.
The only quantity that can be minimized taking into account the overall cost function is T_i. Thus, we want to find a motion law that minimize such quantity. Note that summing up the times the approaching phase and crossing phase requires, we get $t_{tot} = t_i + L_i T_i$, maximize the final velocity of the approaching phase means minimize T_i and since t_i is demanded to the high level problem, we can minimize the time at this stage only maximizing the final velocity $v_{f,i} = v(t_i)$. The simplified model of the vehicle is provided in Equation (4.2). In our problem, every vehicle is distant s_f meters from the conflict zone. The time to arrive t_i is "given" by the MINLP problem. having in mind that minimizing $T_i(t_i)$ is equivalent to maximise its reciprocal $v(t_i)$, we can formulate the optimal control problem:

$$\begin{array}{c} \max_a J = v(t_i) \\ s.t. \\ \text{(4.2)} \\ a_m \leq a \leq a_M \\ v(0) = v_0 \\ s(0) = 0 \qquad s(t_i) = s_f \end{array} \tag{4.6}$$

4.4.2 OCP solution

To solve the problem illustrated in Equation (4.6), we employed a indirect approach using the Pontryagin Minimum Principle (PMP). A summary of the procedure of the Pontriagyn minimum principle is illustrated in [94], at page 69, the overall steps to find the maximum/minimum using the PMP approach are summarized.
We calculate the Pontriagyn Hamiltonian function, since $\partial J/\partial a = 0$, the Hamiltonian contains only the constraints and it is formulated as follows:

$$\mathcal{H} = \lambda_1 v + \lambda_2 a. \tag{4.7}$$

The optimal control $a^\star$ must be a stationary point for the Hamiltonian, To find $a^\star$, we need more informations from the co-state equations and the boundary conditions, that are formulated as:

$$\begin{cases} \dot{\lambda_1}^\star = 0 \\ \dot{\lambda_2}^\star = \lambda_1^\star \end{cases} \tag{4.8}$$

While the boundary conditions are:

$$\begin{cases} s(0) = 0 \qquad s(t_i) = s_f \\ v(0) = v_0 \end{cases} \tag{4.9}$$

Even if we already have initial and final conditions, it is not enought, we must apply this equation taking into

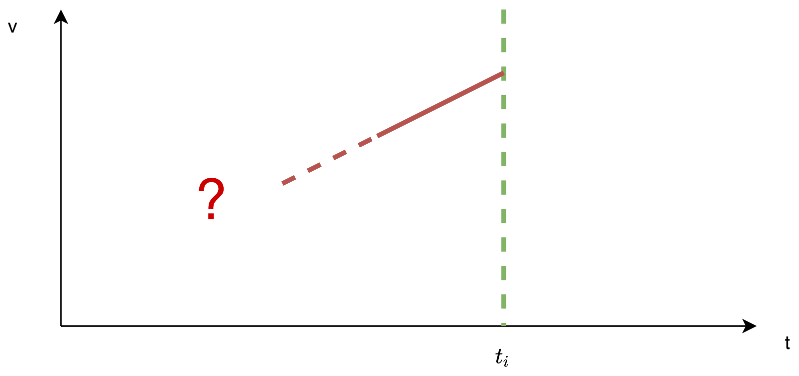


Figure 4.4: Final piece of OCP solution

account possible variation of final time and final state.

$$\left(\mathcal{H}^\star(t_i) + \frac{\partial S}{\partial t_i}\right)\delta t_i + \left(\frac{\partial S}{\partial \boldsymbol{x}} - \boldsymbol{\lambda}^\star(t_i)\right)\delta \boldsymbol{x}(t_i) = 0 \tag{4.10}$$

The variation δt_i is 0, therefore the first part of the equation does not play any role. regarding the final states variations, the variation of the final state is not null: $\delta\boldsymbol{x}(t_i)$. Thus, the derived equations are:

$$\begin{cases} 0 = \lambda_1^\star(t_i) \\ 1 = \lambda_2^\star(t_i) \end{cases} \tag{4.11}$$

Remind that S is the part of the cost function that contains the final states. Now we have all the elements, from (4.8) and (4.11) we can conclude that $\lambda_1^\star = 0 \;\; \forall t$ and $\dot{\lambda_2}^\star = 0 \;\; \forall t$. Taking in consideration this trends of $\lambda^\star$ into the Hamiltonian, we see that to maximize it we need to maximize $\lambda_2 a$. The control a is limited by (4.6). To maximize the product, the sign of the terms must be the same, therefore:

$$\begin{cases} a = a_M \quad (if) \quad \lambda_2^\star > 0 \\ a = a_m \quad (if) \quad \lambda_2^\star < 0 \end{cases} \tag{4.12}$$

Note that $a_m < 0$.
Furthermore we know that the final $\lambda(t_i)$ si equal to 1, so the final control is necessarily a_M. The rest of the control is constant or piecewise constant.

$$a(t_i) = a_M \tag{4.13}$$

At this point we can only conclude that the control present the form in (4.12). In order to understand which is its actual expression in time, we need to rely on some graphical considerations illustrated below.
We know that the final control is maximum acceleration, therefore we expect a shape similar to Figure 4.4.
In order to respect the constraint on the final position $s(t_i) = s_f$, the motion cannot be only an acceleration. At first we must take into account that the position is the area under the curve and that according to (4.12), the control can be only a_m or a_M. Then there exists a time t_{swc} before which we have $a = a_m$ and after this is $a = a_M$. As shown in Figure 4.5, we can have a full acceleration if possible, otherwise we have a switch from min to max acceleration, the third option is to have a stop in between, it will be detailed later. Figure 4.5 shows the three possible options (In the Figure are 4 sice also the case $t = t_{break}$ is shown). Indeed, there is only one value of time t_i such that the motion is only a continous acceleration. Such time is indeed $t_{i,min}$ and it is the time required

to cover the s_f distance with maximum acceleration a_M only. The time $t_{i,min}$ is the lower bound of t_i:

$$t_{i,min} = \frac{v_f - v_0}{a_M}. \tag{4.14}$$

This value is shown in Figure 4.5, where the $v_f(t_i)$ begins.
Vehicles can decelerate up to stop, for obvious reason we cannot plan longitudinal motion where velocity is negative. There exists a value of $t_i = t_{break}$ such that the optimal switching motion decelerate up to negative velocities. After such value all the motions consists in a deceleration up to stop, a time in which vehicle does not move, and then an acceleration up to the final velocity. Figure 4.5 help to understand how the $v_f(t_i)$ function is built, and the corresponding motion for each $v_f(t_i)$.

4.4.3 Velocity limits

In this section we have already introduced the acceleration limits, but we did not discuss the velocity limits. In the approaching phase, for the motion characteristic, the maximum velocity can be only v_0 or v_f. No constraint is introduced on v_0 because of assumption 4.4. The final velocity v_f instead, must be limited taking into account two factors:

The legal speed limit of the road.

$$v_{f,i} \leq v_{lim} \tag{4.15}$$

The curvature of the trajectories. The curvature of the trajectories inside the intersection are used to impose a limit on velocity. Such limit is chosen using 0.3 of the maximum grip in the curve.

$$v_{k,curv} = \left(\frac{\mu g}{|\gamma|}\right)^{\frac{1}{2}} \tag{4.16}$$

For both the listed cases, we impose a limit on v_f. Such a limit is not imposed as a constraint. It is directly imposed in the function $v_f(t_i)$. Since we already have a case-defined function, it is easy to introduce such a limit directly. Furthermore, we take advantage of the piecewise linear approximation to introduce this limit. In particular, we introduce such limits changing the value of $t_{i,\star}$, introduce in Section 4.6.

4.4.4 Final velocity function

The low level problem carries out two outcomes: the motion of the vehicle and the final velocity $v_{f,i}(t_i) = v_i(t_i)$ to be used by the high level problem. To get such expression, we must solve a constant bang bang motion for velocity and position at time t_i.

$$\begin{cases} s_{f,i} = t_i(a_m t_{i,swc} + a_0) - \frac{a_m t_{i,swc}^2}{2} + \frac{1}{2}a_M(t_i - t_{i,swc})^2 \\ v_{f,i} = v_0 + a_m t_{swc} + \frac{1}{2}a_M\left(t_i - t_{swc}\right)^2 \end{cases} \tag{4.17}$$

Where v_0 is the initial velocity, a_m and a_M are the acceleration limits, while t_{swc} is the time to switch from deceleration to acceleration. Solving equation (4.17) for v_f and t_{swc} we get two pairs of solutions which only one have t_{swc} positive. Once the motion is established, the function of the velocity reads as:

$$v_{f,i}(t_i) = a_m t_i + v_0 + \sqrt{(a_m - a_M)\left(2a_m s_f + v_0^2\right)} \tag{4.18}$$

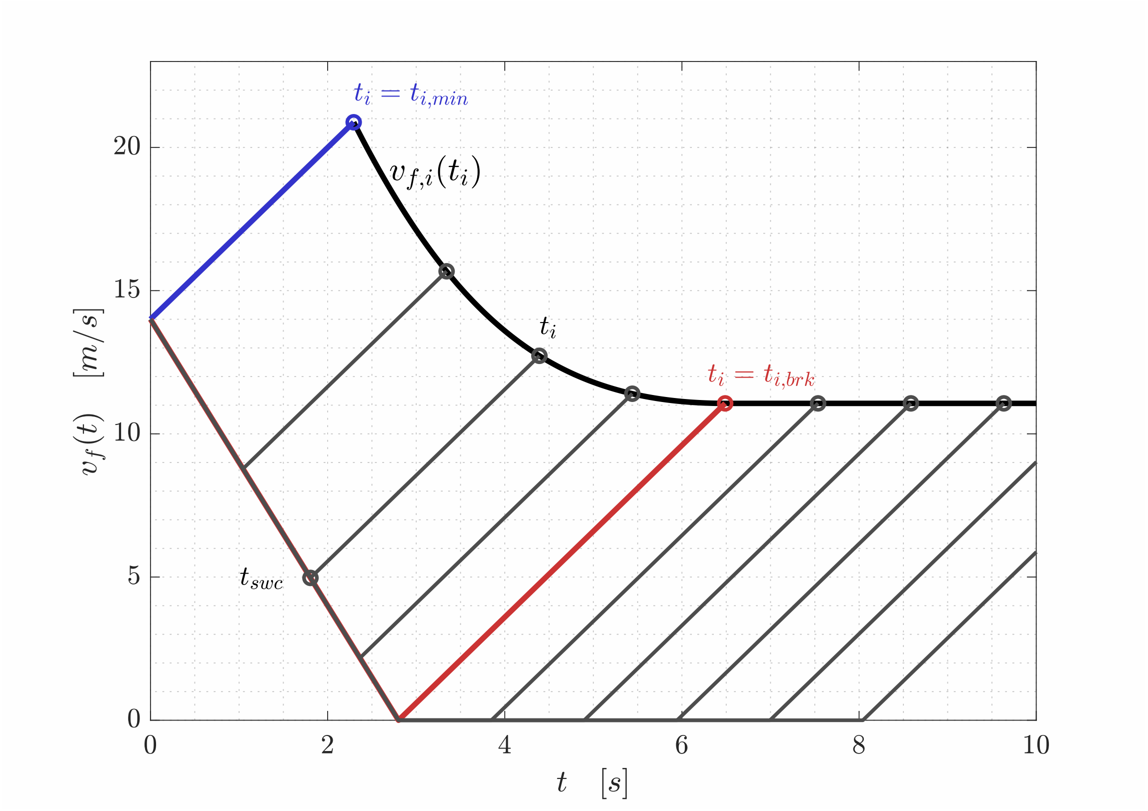


Figure 4.5: Example of $v_f(t_i)$ function construction. In the same plot examples of $v_i(t)$ bang bang motion trajectories are shown. The envelope of the final velocities of the motion is the function $v_f(t_i)$. In Blue and in Red the special cases $t_i = t_{i,min}$, $t_i = t_{i,brk}$

Note that expression (4.20) is valid only for $t_i \in [t_{i,min}, t_{i,\star}]$. The time $t_{i,\star}$ is a time such that the deceleration phase of the vehicle bring it to stop. This means that $t_{i,swc}$ reaches its maximum to stay into positive velocities. The expression for $t_{i,\star}$ reads as:

$$t_{i,\star} = \frac{-a_m a_M v_0 + \sqrt{a_m^3 a_M \left(2a_m s_f + v_0^2\right)}}{a_m^2 a_M} \tag{4.19}$$

For $t_i > t_{i,\star}$ the final velocity is always equal and the function becomes flat. The final function of $v_f(t_i)$, from which $T_i(t_i)$ is derived, read as:

$$v_{f,i}(t_i) = \begin{cases} a_m t_i + v_0 + \sqrt{(a_m - a_M)\, 2a_m s_f + v_0^2} & t \in [t_{i,min}, t_{i,\star}] \\ v_{i,sat} & t > t_{i,\star} \end{cases} \tag{4.20}$$

An example of the function is shown in Figure 4.5.

4.5 high level problem

The core of this chapter is the intersection management problem, which is the high level problem subject to the low level problem. The high level problem concerns the global optimization of the intersection, addressing the minimization of the time required to clear the intersection for all the vehicles, subject to individual vehicle motions

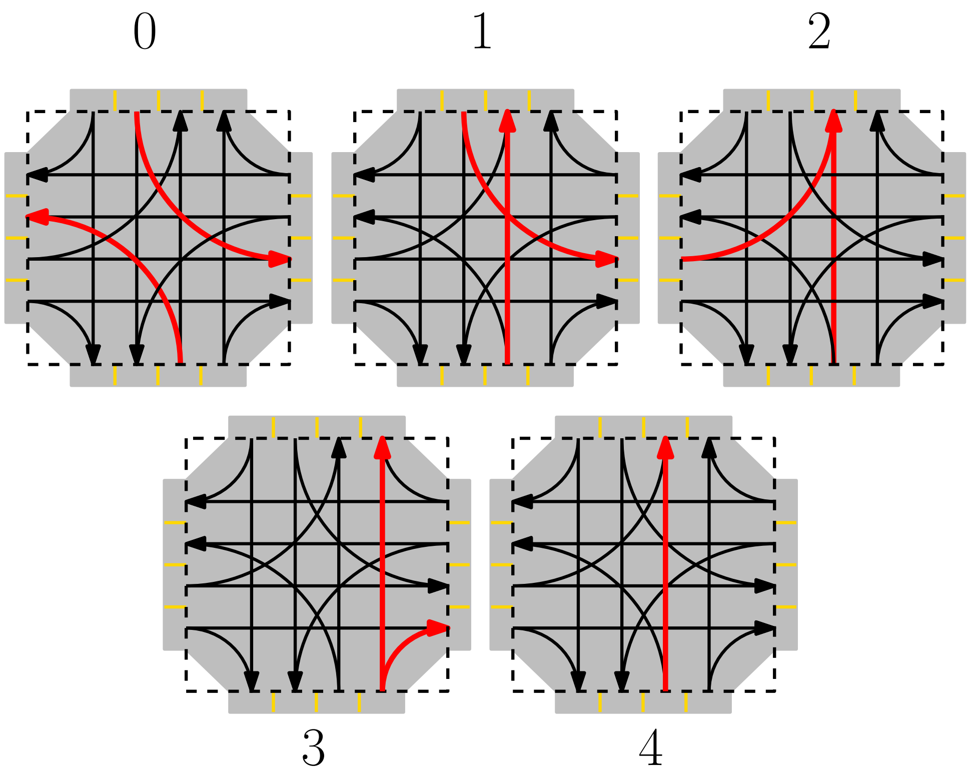


Figure 4.6: Examples of the conflicts between the paths. from left to right one example is provided for each conflict type illustrated in Section (4.1)

from the inner problem. It means it is demanded to find the optimal order and arrival times of all the vehicles involved. The initial states of the vehicles (initial conditions) are given. The problem is to find scheduling of the arrival times t_i that minimizes the overall time elapsed for each vehicle. The degrees of freedom is the order of passage and the time to arrive. The time to cross is a function of the t_i. This function is one of the outcomes of the low level problem.
The scheduling variable determines the crossing order, a scheduling variable concerns the sequence of two vehicles i and j, and it is a binary variable called w_{ij}. If $w_{ij} = 1$ the vehicle i passes first. These variables are needed only when the paths of vehicle i and j present conflicts. Therefore if two vehicles can pass simultaneously, no corresponding scheduling variable is defined.
As stated in Section 4.5.1 the arrival time t_i of the vehicle i is defined as the time at which the vehicle arrives at the conflict zone following its lane. This is the most crucial quantity since it also determines the velocity at which the vehicle i arrives, and therefore the crossing time (see 4.3).

4.5.1 Scheduling Constraints

In an intersection management system, the first concern of the algorithm is definitively the avoidance of collisions. The scheduling constraints (or collision constraints) are aimed at excluding the possibility to have collisions within the CZ. The Scheduling Constraints any pair of vehicles which moving footprints can eventually overlap simultaneously. To understand these constraints, it is important to keep in mind that the velocity is constant inside the Conflict Zone, as stated in Assumption 4.3.
The possible conflicts between various paths are of four types, as stated in the Conflict Matrix definition (see Equation (4.1)) in the cases from 1 to 4. In Figure 4.6 an example of each of them is shown.

Each of this type of conflict is associated with a specific *set of constraints* that may include the scheduling variables and the lengths (See Section 4.2).

The first type of conflicts leads to the following set of scheduling constraints:

$$t_i + L^a_{ij}T_i(t_i) - t_j - L^b_{ji}T_j(t_j) \leq C_{large}(1 - w_{ij}), \tag{4.21a}$$
$$t_j + L^a_{ji}T_j(t_j) - t_i - L^b_{ij}T_i(t_i) \leq C_{large}w_{ij}, \tag{4.21b}$$
$$w_{ij} \in \{0,1\},$$
$$\forall i,j \in \mathcal{V} \quad \text{with} \quad c_{P(i)P(j)} = 1,$$

where $T_i(t_i) = 1/v(t = t_i)$ is the inverse of the constant velocity of the vehicle i in the conflict zone. This quantity is very important and it will appear in most of the constraints. The symbol C_{large} denotes some large positive constant, at least as large as $\max_i \left(t_i + L_{p(i)}T_i\right)$. Selecting is constant is not a problem, one can choose a very large constant. The scope of this constant is to deactivate the constraint when it is multiplied by one (and not by zero). Let us explain how Equation (4.21) works. Given a couple of vehicles $i, j \in \mathcal{V}$ in $m_{c,P(i),P(j)} = 1$, the binary variable w_{ij} indicates the priority, if $w_{ij} = 1$ the vehicle i passes first, on the other hand if $w_{ij} = 0$ the vehicle j passes first. The constraints (4.21) are referred as "either-or" constraints [125] since they are mutually exclusive based on the value of w_{ij}. We can take into account the first constraint (4.21a) assuming $w_{i,j} = 1$.

$$t_i + L^a_{ij}T_i(t_i) \leq t_j + L^b_{ji}T_j(t_j),$$

The left hand side of the inequality represents the total time the priority vehicle i needs to clear the overlapping region. The time t_i it's the time required to reach the CZ, and $L^a_{ij}T_i(t_i)$ is the time to leave the overlapping region of the conflict zone interested in the conflict. The length L^a_{ij} indeed is the length to travel for vehicle i in the CZ to leave the vehicle j free to pass along the path $p(j)$. Vehicle j cannot occupy the ij overlapping region before the time above. To model this, the right hand side is the sum of t_j and $L^b_{ji}T_j(t_j)$ the latter is the additional available time for vehicle j before reaching the overlapping region.

(See Figure 4.6 and Figure 4.1 for visual references of conflicts and lengths respectively). The other inequality (4.21b) represents a constraint that is not active if $w_{ij} = 1$, since the C_{large} makes the inequality always verified. At first glance, this case may seem comprehensive of all the possible conflicts, since one can easily set L^a_{ij} as 0 to model when two vehicles come from the same lane and L^b_{ij} when they travel to the same exit lane. However, in the following cases, we will illustrate that it is not enough.

The next type of conflict models two paths with the exit lane in common but a different entering lane (number 2 in conflict matrix). Such conflicts generate another set of scheduling constraints. The first thing one can notice is than L^b_{ij} and L^b_{ji} by definition are equal to $L_{p(i)}$ and $L_{p(j)}$ respectively. This consideration is not enough to avoid collision since the vehicle may collide in the exit phase. In the exit lanes, the vehicles i and j follow each other. Inside the CZ there will be no collision, but the velocities may end the distances between the vehicles cause a condition at which rear-end collision cannot be avoided. This problem is addressed by introducing a time headway (T^{hdw}_{ij}) between the vehicles that must hold at the exit of the intersection. This time headway can be used as a fixed parameter (but enough to ensure no rear collisions occur), or it can be calculated to keep certain exit conditions in terms of velocity and distance. The scheduling constraints set is then formulated as:

$$t_i + L_{P(i)}T_i(t_i) + T^{hdw}_{ij} - t_j - L^b_{ji}T_j(t_j) \leq C_{large}(1 - w_{ij}), \tag{4.22a}$$
$$t_j + L_{P(j)}T_j(t_j) + T^{hdw}_{ji} - t_i - L^b_{ij}T_i(t_i) \leq C_{large}w_{ij}, \tag{4.22b}$$
$$w_{ij} \in \{0,1\},$$
$$\forall i,j \in \mathcal{V} \quad \text{with} \quad c_{P(i)P(j)} = 2,$$

. An example of a value for T_{hdw} may be required time to reach a maximum velocity v_{max} after exit: s

$$T^{headway}_i(t_i) = \frac{v_{exit} - v_i(t_i)}{a_{M,i}} + T_{gap}, \quad t_i \in I^{t_{i,max}}_{t_{i,min}}, \quad i \in \mathcal{V} \tag{4.23}$$

The velocity v_{max} must be the same for each vehicle per same exit lane, T_{gap} is the time gap (at the velocity of vehicle i) when such vehicle reach the exit of the intersection. This gap is required because if $v_{exit} = v_i(t_i)$ vehicles will overlap in the conflict zone having a rear collision. Note that $T_i(t_i) = 1/v_{f,i}$ is the time to travel one meter in the conflict zone, where the velocity is constant. The third type of conflicts consists of different exit lane but common entering lane. In this case, conversely to the previous, L^b_{ij} is equal to zero since in the conflict zone the paths starts in the same point. In this case the order of the vehicle is known in advance. Since they are in the same lane, the leading vehicle always pass before the other. Which of the two is the leading vehicle can be easily retrieved from the curvilinear coordinates $s_i(t)$. The corresponding set of constraints reads as:

$$t_i + L^a_{i,j} T_i(t_i) + T^{hdw}_{ij} - t_j \leq 0, \tag{4.24a}$$

$$\forall i, j \in \mathcal{V} \quad \text{with} \quad m_{c,p(i),p(j)} = 3 \wedge s_i > s_j, \tag{4.24b}$$

Also, in this case, a T^{hdw}_{ij} can avoid rear collisions in the vicinity of the CZ, but it is not enough to ensure no rear collisions occurs before the intersection. We did not find an expression of T^{hdw}_{ij} that ensures such condition. Rear collisions in the same lane are detailed later on (See (4.26)).
The last type of collision occurs when the paths coincide. This case essentially corresponds to having two vehicles that are traveling on the same lane even in the CZ. Between these two vehicles, only rear collisions may occur. Note that type 4 is always the diagonal of the conflict matrix. This conflict may be modelled in a "follow leading vehicle" fashion. But since the motion is heterogeneous from out ot in the CZ, we must take into account that the velocity amy be different and we must use the following condition:

$$\begin{gathered} t_i + L_{P(i)} \left(T_i(t_i) - T_j(t_j) \right) + T^{hdw}_i(t_i) - t_j \leq 0, \\ \forall i, j \in \mathcal{V} \quad \text{with} \quad m_{c,p(i),p(j)} = 4 \wedge s_i > s_j, \end{gathered} \tag{4.25a}$$

This condition contains the already discussed time headway T^{hdw}, that is ensured to be respected at the moment the first vehicles leaves the intersection. Indeed if we remove T^{hdw} and we expand the expression in Equation (4.25) we get:

$$t_i + L_{p(i)} T_i(t_i) \geq t_j + L_{p(j)} T_j(t_j)$$

that essentially constraint the following vehicle to arrive at the end of the intersection at most at the same time ofashe first, we therefore add the time headway to avoid arriving at the same time. Note that $T_i(t_i) = 1/v_{f,i}$ is the inverse of the constant velocity of the vehicle i in the conflict zone thus it is the time to travel one meter in the conflict zone. During the approaching phase, the motion provided by the OCP does not ensure that there will be no rear collisions. The outer scheduling problem and the inner problem interface to each other only through the $T_i(t_i)$ function and the t_{start} constraints. To avoid this type of constraints during the approaching phase, we must introduce another constraint.
The approaching phase motion presents a time $t_{swc,i} \in [0, t_i)$ in which the vehicle starts to accelerate, if this time is 0 the vehicle accelerates only, while the $t_{swc} = t_i$ is not considered since it is a limit case for the initial conditions, and it will contradict Assumption 4.4.

$$t_{swc,i}(t_i) := \frac{v_{0,i} - v_i(t_i) + a_{M,i} t_i}{a_{M,i} - a_{m,i}}, \quad t_i \in I^{t_{i,max}}_{t_{i,min}}, \quad i \in \mathcal{V}. \tag{4.26}$$

Complementary to the time headway introduced in Equation (4.25), a new constraint is added to avoid rear collisions:

$$\begin{gathered} T^{switch}_i(t_i) - T^{switch}_j(t_j) \leq 0, \\ \forall i \in \mathcal{V} \quad \text{with} \quad c_{P(i)P(j)} = 4. \end{gathered} \tag{4.27}$$

The constraints are now defined. However, a last aspect must be pointed out before formulating the final MINLP problem. The total time elapsed for all the vehicles to clear the intersection is not the sum of the time to leave the intersection. Instead, it is the maximum. However, in the optimization problem, we cannot use the maximum since some of the vehicles involved do not affect the time to clear the last vehicle therefore we must use the sum to minimize the time to clear each vehicle involved in the intersection. The final MINLP is the following:

$$\begin{aligned} \min_{t_i, w_{ij}} \quad & \sum_{i=1}^{N} \left[t_i + L_{P(i)} T_i(t_i) \right], \\ \text{s.t.} \quad & (4.21)(4.22)(4.24a)(4.25)(4.27)(4.14) \end{aligned} \tag{4.28}$$

The approach we decided to use to solve this problem is to first approximate the problem into a linear one (bounding the error) and then to solve with commercial linear solvers that are reliable and fast. We will see that state-of-the-art linear solver such as GUROBI MILP-solver [58] are fast enough to handle linearized versions of the problems in a reasonable time for applications. How the problem is linearized is shown in the next Section.

4.6 Piecewise linearization

The high level problem presents two functions of continous variable t_i that are non-linear: $T_i(t_i)$ and $v_f(t_i)$. These two function are reciprocal each other, indeed $T_i(t_i) = 1/v_f(t_i)$. We hereby show the linearization technique used to find an approximated MILP to model the MINLP presented in 4.5. The problem is reformulated into a linear one using the piecewise linearizations of the nonlinearities $T_i(t_i)$ and $v_f(t_i)$, on some discretizations of the intervals $[t_{i,min}, t_{i,max}]$. Where $t_{i,max}$ denotes some reasonable time at which we are sure that the i's car will enter an intersection. We will se that we can use an arbitrary large value for $t_{i,max}$, and for this reason we use the same $t_{i,max} = t_{max}$ for all $i \in \mathcal{V}$, we cannot do the same for $t_{i,min}$ since it is different for every evhicle and it comes from the lowe level problem (see Section 4.4).

The solvers do not directly handle piecewise linear functions. We need to define them as some constrained combination of linear functions and integer variables. The technique we use to model such functions is used in [125].

4.6.1 Piecewise linearization technique

The piecewise linearization, according to [125] consists in modeling every segment of the piecewise linear function as a convex combination of the two breakpoints values. Which segment to choose is modeled by binary variables that are constrained to be nonzero only for one segment at a time. The new variables and constraints are illustrated in (4.29) and discussed below. For now, we only consider the function $T_i(t_i)$ to explain the method, and then we discuss how we actually model the functions. For each $i \in \mathcal{V}$ we have

$$t_i = \sum_{k=1}^{K_i} \lambda_{i,k} \, t_{i,k}, \tag{4.29a}$$

$$T_{i,pw} = \sum_{k=1}^{K_i} \lambda_{i,k} \, T_i(t_{i,k}), \tag{4.29b}$$

$$\sum_{k=1}^{K_i} \lambda_{i,k} = 1, \quad \sum_{k=1}^{K_i - 1} z_{i,k} = 1, \tag{4.29c}$$

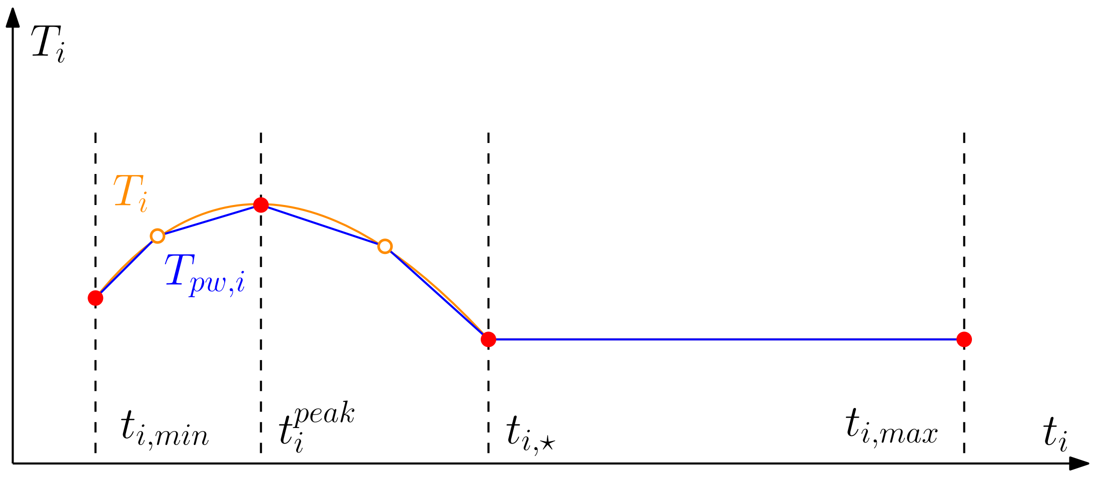


Figure 4.7: Example of the piecewise linearization of $T_i(t_i)$ function, $t_{i,\star}$ is known. Also the stationary point $t_{i,peak}$ can be calculated in advance if exists.

$$\begin{array}{rcl} \lambda_{i,1} & \leq & z_1, \\ \lambda_{i,k} & \leq & z_{i,k} + z_{i,k-1}, \quad k \in \{2, \dots, K_i - 1\}, \\ \lambda_{i,K_i} & \leq & z_{K_i - 1}, \end{array} \tag{4.29d}$$

$$\begin{array}{rcl} \lambda_{i,k} & \geq & 0, \quad k \in \{1, \dots, K_i\}, \\ z_{i,k} & \in & \{0, 1\}, \quad k \in \{1, \dots, K_i - 1\}, \end{array} \tag{4.29e}$$

where $t_{i,k}$, $k \in \{1, \dots, K_i\}$, are the breakpoints of the argument t_i in the interval $[t_{i,min}, t_{i,max}]$ with $t_{i,1} = t_{i,min}$ and $t_{i,K_i} = t_{i,max}$, K_i its an integer constant that denotes the number of discretization points of the $i-th$ time interval. Note that the distance between the breakpoints can vary between segments. Let us now explain the new variables presented in (4.29). The real variables lambdas $\lambda_{i,k}$, $k \in \{1, \dots, K_i\}$ plays the role of weighting the two values of the breakpoints of the function on the selected segment. The binary variables (z-variables) $z_{i,k}$, $k \in \{1, \dots, K_i - 1\}$ indicate the segment on which the function is evaluated. The constraint on the sum of the binary variables (4.29c) guarantees that only one value along k is equal to one and the other are equal to zero (this is to select only one segment at a time). In the same Equation (4.29c) we find that the sum of lambdas is 1, but thanks to (4.29d) two values of lambda are non-zero, and they correspond to the weight of the breakpoints of the convex combination modeling the select segment. At the end (4.29a) and (4.29b) are linear combinations over the t_i corresponding segment.

4.6.2 velocity function piecewise linearization

As stated earlier, we have two non-linear functions: $T_i(t_i)$ and $v_f(t_i)$, one is the reciprocal of the other. Since the argument is the same, we do not need to introduce a new set of Equations (4.29), we just need to add the piecewise linear function using the piecewise arguments already defined. Thus, we need to introduce only the following equation:

$$v_{f,i,pw} = \sum_{k=1}^{K_i} \lambda_{i,k} \, v_{f,i}(t_{i,k}), \tag{4.30}$$

4.6.3 Breakpoints selection

Our method to approximate the function is to select some breakpoints on the *non-linear* function and to use them to build a piecewise linear function that approximates the non-linear one. As one can imagine, the choice of such breakpoints is crucial to get the best out of the piecewise function in terms of the accuracy of the approximation. In Figure 4.7 an example of a choice for breakpoints is shown. In the case of the function illustrated in low level problem Section, we can exploit the structure of the function to select effective breakpoints. The red points in Figure 4.7 represent 4 (or 3) breakpoints that are always selected: the first is $t_{i,min}$ that is already discussed. The second is a stationary point in the non-linear function, $t_{i,peak}$ that does not always exist. The third point is the point at which the final velocity of the function saturates since the vehicle has to stop after the deceleration phase. The last point is $t_{i,max}$. The other breakpoint can be placed equidistantly between $t_{i,min}$ and $t_{i,peak}$ and between $t_{i,peak}$ and $t_{i,\star}$. If $t_{i,peak}$ is not defined the breakpoints can distributed between $t_{i,min}$ and $t_{i,\star}$. The last part of the functions, from $t_{i,\star}$ to $t_{i,max}$ is flat, therefore there is no need for further points. The error is zero since the true function, and the piecewise linear function coincide in such range.

4.6.4 final MILP

The final problem is then obtained using the technique illustrated in Section 4.6 to approximate the non-linear problem in Equation (4.28). The final Mixed Integer Linear Problem reads ad:

$$\begin{gathered}\min_{t_i, w_{ij}} \sum_{i=1}^{N} \left(t_i + L_{P(i)} T_{pw,i}\right) \\ \text{s.t.} \qquad . \\ (4.21)(4.22)(4.24a)(4.25)(4.27)(4.29)(4.14)\end{gathered} \tag{4.31}$$

The problem can be solved by commercial solvers in a reasonable time. We will discuss this aspect in the next Section.

4.7 Simulations and results

In order to validate the proposed algorithm, we performed simulations in a complex intersection presenting 16-lanes, 8 entering lanes, and 8 exit lanes. The geometry of the intersection is shown in 4.8. We test our framework with 32 vehicles, 4 per entering lane, the vehicle distances are all 20 meters while the velocity is randomly generated around 50 KM/h ($\approx$ 13.9 m/s). The distribution for the velocity is $\mathcal{N}(13.9, 1)$.
In Table 4.2 an example of the results is shown. In Figure 4.8 the trajectories are shown. The same color is used for the same path and similar color for the same entering lane (for example, two shades of blue for paths 1 and 2 that share the same entering lane). When an external limit gives the final velocity, the deceleration is the same as reaching the maximum velocity up to the time switch. After such time the motion is realized with a third polynomial (since in such case we do not need to use the maximum and minimum acceleration) that fits final velocity, time, and position at t_i.

4.7.1 Comparison with distance-based heuristic

To prove the benefit of the proposed framework, we made a comparison with a First-Arrive-First-Served approach. The latter is simulated with another bi-level optimization problem. Which is a different version of the proposed approach. This approach consists in applying the simple rule that the closer the vehicle the first it has the right to pass.

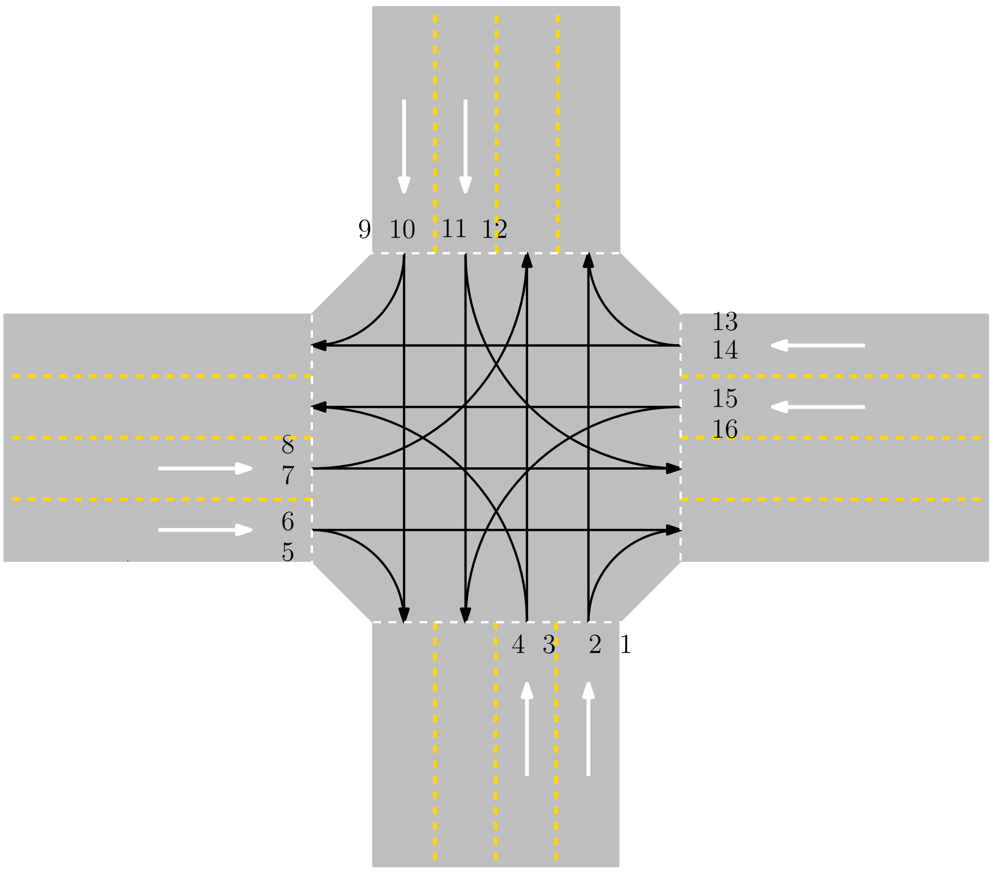


Figure 4.8: Geometry of 16-lanes intersection, all the 16 paths are shown. Every lane has two possible paths, the intersection is symmetric.

n	p_i	$s_{i,0}$	$v_{i,0}$	t_i	T_i
1	1	50 m	14.44 m/s	3.61 s	0.158s
2	2	70 m	15.73 m/s	10.55 s	0.061s
3	1	90 m	11.64 m/s	11.63 s	0.158s
4	2	110 m	14.76 m/s	14.48 s	0.059s
5	3	50 m	14.22 m/s	4.08 s	0.059s
6	4	70 m	12.59 m/s	10.78 s	0.131s
7	3	90 m	13.47 m/s	14.27 s	0.059s
8	4	110 m	14.24 m/s	15.24 s	0.131s
9	5	50 m	17.48 m/s	1.47 s	0.158s
10	6	70 m	16.67 m/s	6.48 s	0.059s
11	5	90 m	12.55 m/s	11.10 s	0.158s
12	6	110 m	16.93 m/s	13.40 s	0.059s
13	7	50 m	14.63 m/s	5.23 s	0.070s
14	8	70 m	13.84 m/s	6.47 s	0.131s
15	7	90 m	14.61 m/s	9.34 s	0.059s
16	8	110 m	13.70 m/s	18.20 s	0.131s
17	9	50 m	13.78 m/s	4.04 s	0.158s
18	10	70 m	15.39 m/s	7.78 s	0.060s
19	9	90 m	15.31 m/s	10.74 s	0.158s
20	10	110 m	15.32 m/s	14.29 s	0.059s
21	11	50 m	14.57 m/s	4.03 s	0.059s
22	12	70 m	12.69 m/s	11.61 s	0.131s
23	11	90 m	14.62 m/s	14.09 s	0.059s
24	12	110 m	15.53 m/s	15.51 s	0.131s
25	13	50 m	14.39 m/s	4.61 s	0.158s
26	14	70 m	14.93 m/s	8.54 s	0.059s
27	13	90 m	14.63 m/s	9.79 s	0.158s
28	14	110 m	13.60 m/s	14.12 s	0.059s
29	15	50 m	14.19 m/s	4.72 s	0.065s
30	16	70 m	13.11 m/s	6.82 s	0.131s
31	15	90 m	14.79 m/s	9.39 s	0.059s
32	16	110 m	12.75 m/s	18.47 s	0.131s

Table 4.2: Solution example: initial conditions, paths and results of the scheduling

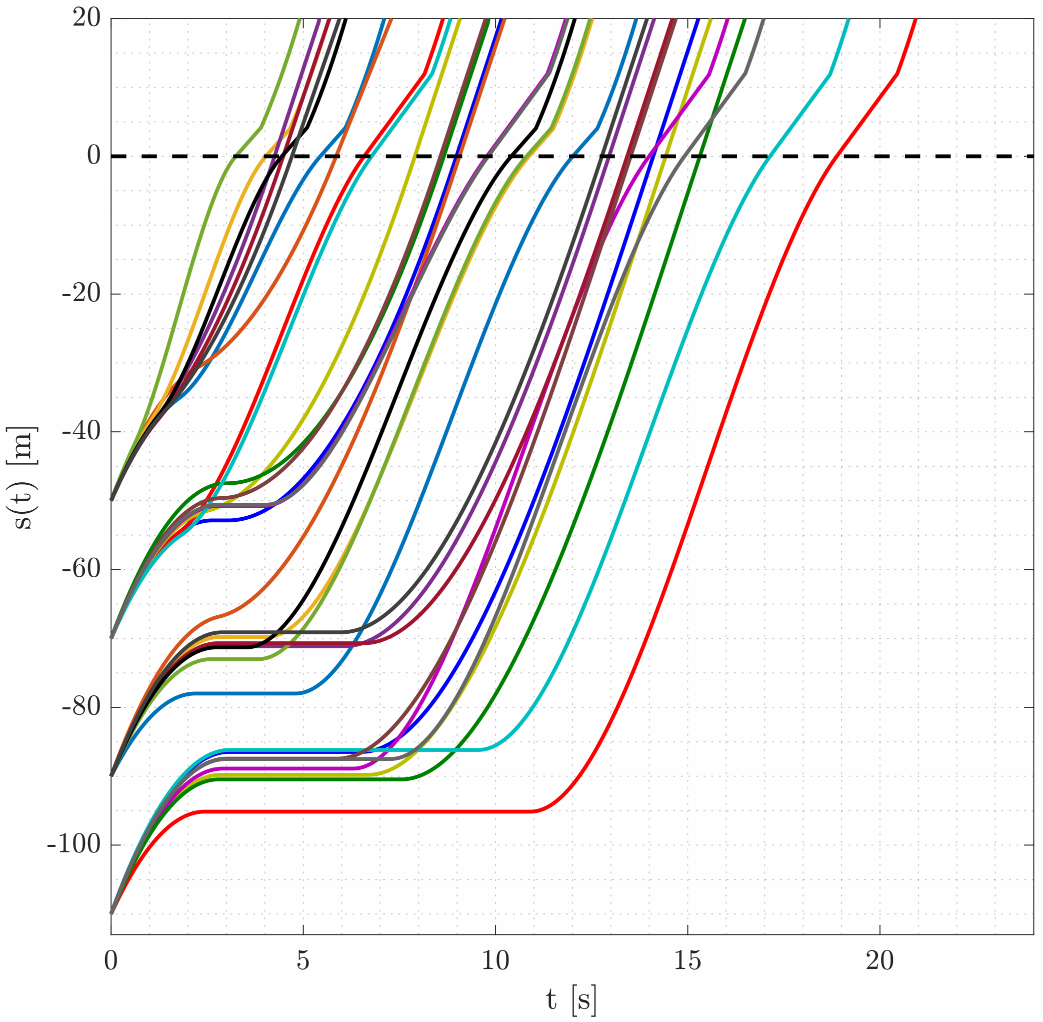


Figure 4.9: Trajectories of the vehicle along the curvilinear coordinate s, same colors are used for same path and similar colors for same entering lane

To get the order of vehicles, we apply the following Linear Problem:

$$\min_{t_i} \sum_{i=1}^{N} \left(t_i + L_{P(i)} T_{pw,i}\right)$$

$$\text{s.t.}$$

$$(4.21) \begin{cases} w_{ij} = 1 & s_i \leq s_j \\ w_{ij} = 0 & s_i > s_j \end{cases} . \tag{4.32}$$

$$(4.22) \begin{cases} w_{ij} = 1 & s_i \leq s_j \\ w_{ij} = 0 & s_i > s_j \end{cases}$$

$$(4.24a)(4.25)(4.27)(4.29)$$

Let us now discuss the difference between the problem just defined in (4.32), and the framework's proposed problem in (4.31). The first is a linear problem since we removed the binary variables w_{ij}. The order now is pre-defined by the distances. The constraints (4.21) and(4.22) in (4.31) where composed of two inequalities each, that activates and deactivates basing on the values of the binary values. In problem (4.32) the active inequality is the one associated with the closest vehicle.
For instance let us take the constraint (4.21) and two vehicles i and j such that $s_i < s_j$, w_{ij} in this case is constant equal to 1, and in (4.21) only the inequality (4.21a) is actually active, having the one with the null right-hand side.
In Table 4.3 the results applying the FAFS framework. Note that this approach is still optimal in term of leaving times of each vehicle, in only the order that is predefined by the positions.
As a comparison term, we use the objective function. It is the sum of the times all the vehicles takes to exit the intersection expressed by $\sum_{i=1}^{N} \left[t_i + L_{P(i)} T_i(t_i)\right]$. The objective function for the proposed framework is 440.5 seconds, and the FAFS approach obtain 659.6 seconds. The improvement is 33.7 %. The result is promising since the FAFS comparison approach was optimal as well since the time is minimized, only the order was fixed by the distances of the vehicles.

4.8 Additional observations

To understand the difference between our approach and the FIFO case, we show a plot that we called xyz or "space-time trajectory plot" In Figure 4.11. Such plot uses 3 axis, x, y and t respectively. It shows the trajectories of vehicles in terms of $x(t)$ and $y(t)$ on the horizontal axes, while the vertical axis is the time t. The trajectories do not touch on those plots if there are no collisions (net of vehicle dimensions). The space-time trajectory plot of the proposed approach is shown in Figure 4.11a, in direct comparison with Figure 4.11b that shows the same plot of the FAFS approach results.
It is evident from the plots that the optimized approach tends to make better use of the space and time volume. The height of the occupied volume is the total time elapsed. Such quantity is improved by 44%, cutting the total time elapsed in almost a half. The FAFS approach results in a total time elapsed of 50.4 seconds while using the proposed approach. Such time reduces to 28.6 seconds.
Zooming some regions of the xyt plots, we can recognize some patterns that explain such a difference of time. Two examples are now listed and detailed.

Optimally filling space-time slots When autonomous vehicles can optimize their order instead of following a rigid rule (incomplete safety), some "missed opportunities" of passing for some vehicles arise. To show this advantage visually when passing from FAFS to an optimal order, the xyt diagram helps. In Figure 4.12, we show the xyz trajectories and time of the first vehicles that pass in both experiments (blue for optimal and red for FAFS). It is evident how the last two vehicles (the two higher, almost straights trajectories in red) in the First-Arrive-First-Served could have passed before, and this is what happens in the optimal framework,

n	p_i	$s_{i,0}$	$v_{i,0}$	t_i	T_i
1	1	50 m	12.39 m/s	4.90 s	0.158s
2	2	70 m	13.46 m/s	13.38 s	0.059s
3	1	90 m	13.74 m/s	15.55 s	0.158s
4	2	110 m	14.18 m/s	31.34 s	0.059s
5	3	60 m	13.64 m/s	5.77 s	0.059s
6	4	80 m	14.34 m/s	18.16 s	0.131s
7	3	100 m	14.29 m/s	26.81 s	0.059s
8	4	120 m	12.65 m/s	36.11 s	0.131s
9	5	50 m	12.95 m/s	4.55 s	0.158s
10	6	70 m	13.16 m/s	12.30 s	0.059s
11	5	90 m	13.39 m/s	13.40 s	0.158s
12	6	110 m	13.58 m/s	30.26 s	0.059s
13	7	65 m	13.91 m/s	7.01 s	0.059s
14	8	85 m	10.87 m/s	21.12 s	0.131s
15	7	105 m	13.44 m/s	28.05 s	0.059s
16	8	125 m	15.14 m/s	39.08 s	0.131s
17	9	50 m	12.83 m/s	4.62 s	0.158s
18	10	70 m	14.83 m/s	11.22 s	0.059s
19	9	90 m	14.25 m/s	12.30 s	0.158s
20	10	110 m	13.87 m/s	28.77 s	0.059s
21	11	70 m	14.08 m/s	11.01 s	0.059s
22	12	90 m	12.33 m/s	24.08 s	0.131s
23	11	110 m	13.82 m/s	29.39 s	0.059s
24	12	130 m	15.50 m/s	42.04 s	0.131s
25	13	50 m	14.00 m/s	3.90 s	0.158s
26	14	70 m	13.94 m/s	10.14 s	0.059s
27	13	90 m	13.17 m/s	14.55 s	0.158s
28	14	110 m	13.87 m/s	27.69 s	0.059s
29	15	55 m	14.13 m/s	4.77 s	0.059s
30	16	75 m	14.33 m/s	14.20 s	0.131s
31	15	95 m	13.53 m/s	26.37 s	0.059s
32	16	115 m	13.66 m/s	32.15 s	0.131s

Table 4.3: Solution example: initial conditions, paths and results of the scheduling when the problem is the one applying the F.A.F.S. approach.

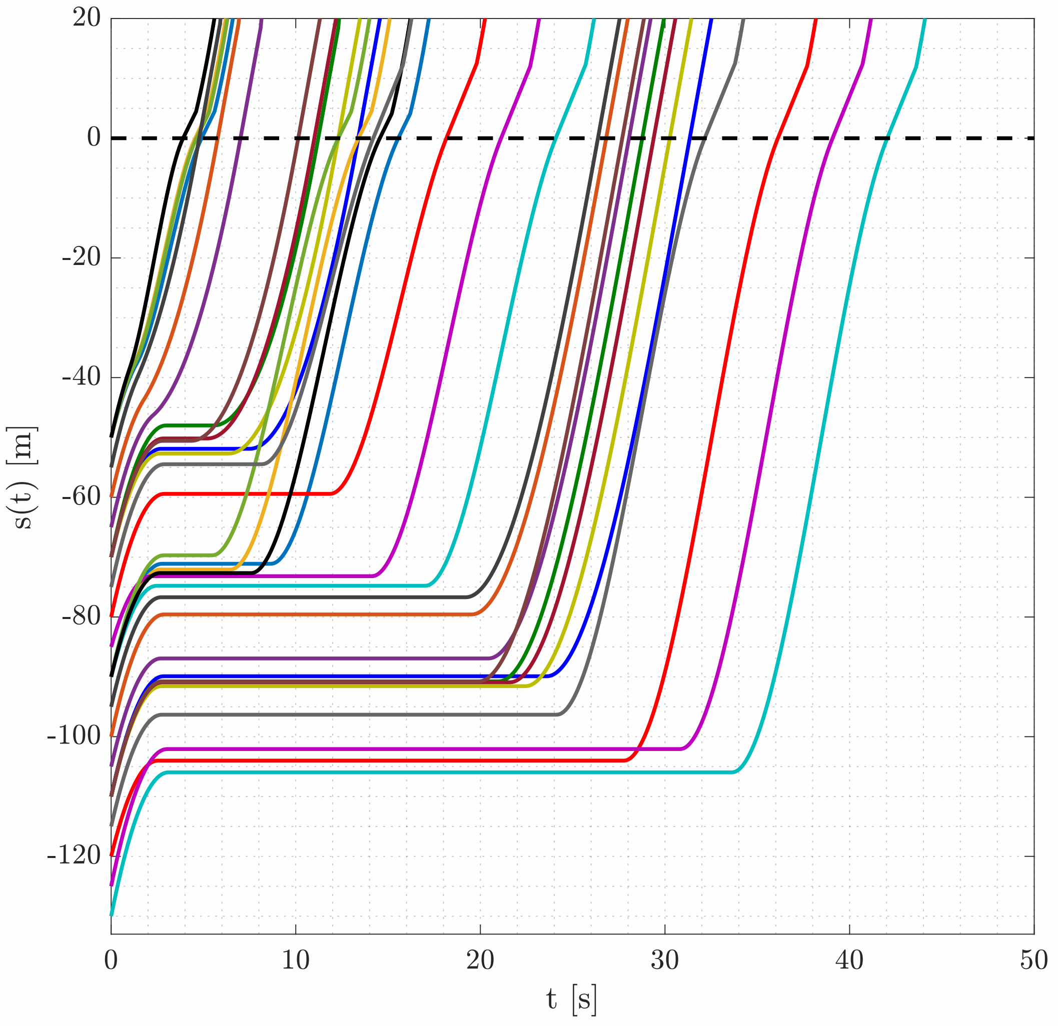
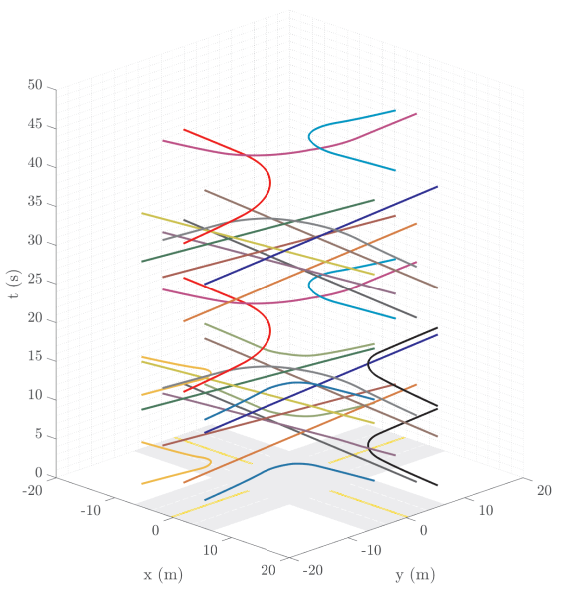
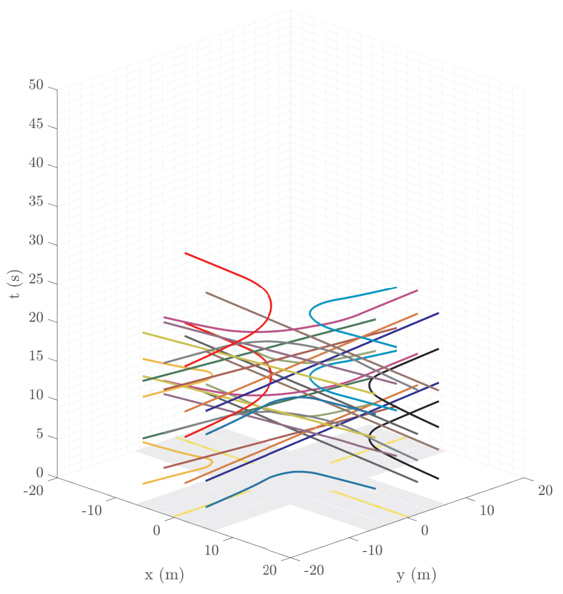


Figure 4.10: Trajectories of the vehicle along the curvilinear coordinate s, same colors are used for same path and similar colors for same entering lane



(a)



(b)

Figure 4.11: Trajectories of the vehicles in time, on the left (a) the results of the FAFS approach, on the right (b), the results using the proposed approach. The proposed approach fill the space-time volume occupying all the gaps.

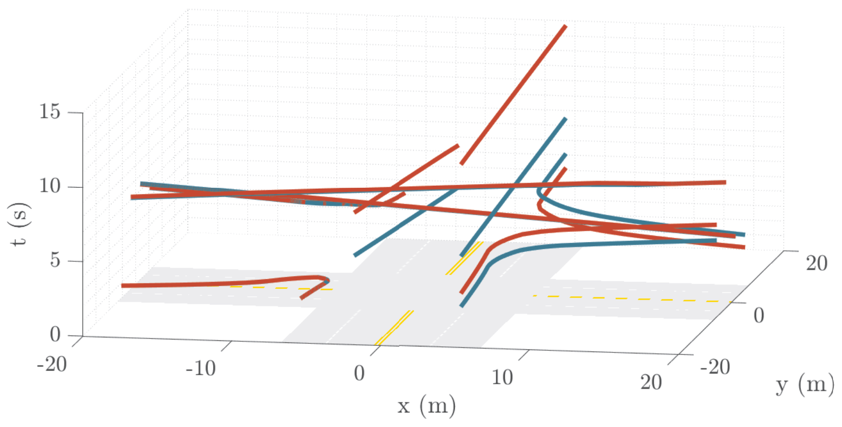


Figure 4.12: Trajectories of the vehicles in a 3d plot with the time axis (xyt plots). Solutions of the first 6 vehicles for the proposed approach (in blue) and the FAFS approach.

where they pass. The two corresponding xyt trajectories that improve the overall timing passing before are indicated in the plot. Interestingly, with this change, the xyt trajectory of the other vehicles did not change much, showing that there was an almost empty "space-time" slot to take advantage of. Looking at Figure 4.12, we see that the xyz trajectories make the best use of the volume of the plot. This is visible also in the comparison between Figure 4.11a and Figure 4.11b. Not that the height of the occupied volume is the quantity that we optimize.

Letting someone else pass We can recognize behavior expected in human driving from the first vehicles. The vehicle coming from the south (vehicle 1 in Figure 4.12) has the right of way on the vehicle coming from the west (vehicle 2). In the FAFS case (in red), vehicle 1 takes precedence, forcing vehicle 2 to slow down a lot. In the optimization case, vehicle 1, even being closer to the intersection, considers vehicle 2 and gives way to it. From the plot, vehicle 1 lost around 1 second, but vehicle 2 gained 5 seconds. This kind of behavior can also be seen in everyday driving (if we are not selfish) when we consider the velocities of our vehicles to decide to give way or not, minimizing the overall "wasted" time.

4.8.1 Conclusions and future developments

We proposed a bi-level optimization algorithm for intersection management for CAV whose scope is to minimize time. We considered a simple OCP with a simple vehicle model on the lower level for each vehicle. On upper-level consists of an MINLP where non-linearities are approximated (with bounded error) with a piecewise linearization techniques. w.r.t. all the possible conflicts are treated. The comparison with a partial optimization on a popular heuristic (First Arrived First Served) shows that the framework outperforms such heuristic by a significant amount (>30%), even if the heuristic's times are optimized. The overall time elapsed presents even a better improvement, that is 44%. This difference is due to the difference between the cost function and the total time elapsed. The cost function is the sum of the overall time taken from each vehicle. In the future, this work may benefit from a consistent amount of simulations sufficient to perform a statistical analysis and to retrieve heuristics that are more applicable in industrial use cases.
This work can be extended in several ways: implementing more sophisticated vehicle models, testing the algorithm's limits with an increasing number of vehicles, introducing uncertainty in states, and studying the robustness.

Glossary

ADAS Advanced Driving Assistance Systems. 2, 3, 7

CAV Connected Autonomous Vehicle. 2, 127

CD-MPC Co-Driver prediction MPC. 9, 87, 89, 92, 93, 95, 96, 97, 98

CSS Constrained Space Stop. 3, 7, 10, 14, 15, 16, 17, 18, 28, 35

CST Constrained Space Time. 3, 7, 16, 18, 19, 28

CZ Conflict Zone. 49, 103, 107, 113, 114, 115

ECMS Equivalent Consumption Minimization Strategie. 64

eDLM enhanced Dynamic Local Map. 21

EMS Energy Management Sistem. 8, 63, 64, 65, 66, 72, 80, 81, 83, 85, 86

EP-MPC Exact Prediction MPC. 86, 87, 89, 92, 93, 95

FAFS First-Arrive-First-Served. 118, 122, 125, 126, 127

FFCT Free Flow Constrained Time. 3, 11, 12, 13, 15, 17, 69

FSP Fuel Saving Percentage. 9, 88, 89, 92, 93

HEV Hybrid Electric Vehicle. 2, 63, 64, 75, 82, 83

HMI Human Machine Interface. 2, 4, 21, 24

ICE internal combustion engine. 64, 75, 81, 88

MINLP Mixed Integer Non-Linear Programming. 2, 4, 101

MINLP Mixed Integer Linear Programming. 101

MPC Model Predictive Control. 2, 4, 5, 8, 63, 64, 65, 83, 85, 86, 87, 93, 129

MSPRT Multi Hyphothesis Sequential Ratio Test. 67, 68, 70

NLP Non-Linear optimization Problem. 84, 85

OCP Optimal Control Problem. 4, 65, 66, 81, 82, 83, 84, 85, 86, 87, 101, 115, 129

OF-OCP off-line OCP. 86, 87, 92, 93, 95, 97, 98

ORU On Road Unit. 23

PMP Pontryagin Minimum Principle. 64, 109

PTW Power Two-Wheeled. 3

RB Rule Based. 64

RCLP Relative Capacity Loss Percentage. 4, 9, 93, 95

RSU Road Side Unit. 24

SOC State-Of-Charge. 65, 82, 83, 87

TTC Time To Collision. 57

V2X Vehicle-To-Everything. 2, 3, 59

VRU Vulnerable Road User. 7, 4, 26, 57

WWD Wrong Way Driving. 26, 57

Bibliography

[1] *Care: The european union's road accident database*. Accessed: 2021-11-11.

[2] *E.u. trace project*. https://cordis.europa.eu/project/id/027763. Accessed: 2021-11-11.

[3] *Interactive*. `https://www.interactive-ip.eu/`. Accessed: 2020-06-10.

[4] *Motorcycle accident in-depth study (maids)*. Accessed: 2021-11-11.

[5] *Safestrip safe and green sensor technologies for self-explaining and forgiving road interactive applications*. https://cordis.europa.eu/project/id/723211. Accessed: 2021-11-11.

[6] *Motorways 2018*, 2018.

[7] *Traffic safety pedestrians basic facts 2018*, 2018.

[8] M. Ahmane, A. Abbas-Turki, F. Perronnet, J. Wu, A. El Moudni, J. Buisson, and R. Zeo, *Modeling and controlling an isolated urban intersection based on cooperative vehicles*, Transportation Research Part C: Emerging Technologies, 28 (2013), pp. 44–62.

[9] F. Allgöwer and A. Zheng, *Nonlinear model predictive control*, vol. 26, Birkhäuser, 2012.

[10] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, *Driver behavior classification at intersections and validation on large naturalistic data set*, IEEE Transactions on Intelligent Transportation Systems, 13 (2012), pp. 724–736.

[11] I. Awolusi and E. D. Marks, *Active work zone safety: preventing accidents using intrusion sensing technologies*, Frontiers in built environment, 5 (2019), p. 21.

[12] J. E. Bakaba and J. Ortlepp, *Safety of characteristic subsections of roadwork zones on motorways*, Transportation Research Procedia, 15 (2016), pp. 283–294. International Symposium on Enhancing Highway Performance (ISEHP), June 14-16, 2016, Berlin.

[13] C. W. Baum and V. V. Veeravalli, *A sequential procedure for multihypothesis testing*, IEEE Transactions on Information Theory, 40 (1994).

[14] P. M. Bays and D. M. Wolpert, *Computational principles of sensorimotor control that minimize uncertainty and variability*, The Journal of physiology, 578 (2007), pp. 387–396.

[15] C. Bila, F. Sivrikaya, M. Khan, and S. Albayrak, *Vehicles of the future: A survey of research on safety issues*, IEEE Transactions on Intelligent Transportation Systems, PP (2016), pp. 1–20.

[16] F. Biral, G. Valenti, E. Bertolazzi, and A. Steccanella, *Cooperative safety applications for c-its equipped and non-equipped vehicles supported by an extended local dynamic map built on safe strip technology*, 05 2019, pp. 733–740.

[17] A. Bisoffi, F. Biral, M. Da Lio, and L. Zaccarian, *Longitudinal jerk estimation for identification of driver intention*, in 2015 IEEE 18th International Conference on Intelligent Transportation Systems, IEEE, 2015, pp. 1855–1861.

[18] ——, *Longitudinal jerk estimation of driver intentions for advanced driver assistance systems*, IEEE/ASME Transactions on Mechatronics, 22 (2017), pp. 1531–1541.

[19] A. Biswas and A. Emadi, *Energy management systems for electrified powertrains: State-of-the-art review and future trends*, IEEE Transactions on Vehicular Technology, 68 (2019), pp. 6453–6467.

[20] R. Bogacz and K. Gurney, *The basal ganglia and cortex implement optimal decision making between alternative actions*, Neural computation, 19 (2007), pp. 442–477.

[21] H. Borhan and A. Vahidi, *Model predictive control of a power-split hybrid electric vehicle with combined battery and ultracapacitor energy storage*, vol. 1, 08 2010, pp. 5031 – 5036.

[22] H. A. Borhan, A. Vahidi, A. M. Phillips, M. L. Kuang, and I. V. Kolmanovsky, *Predictive energy management of a power-split hybrid electric vehicle*, in 2009 American Control Conference, 2009, pp. 3970–3976.

[23] P. Bosetti, M. Da Lio, and A. Saroldi, *On the human control of vehicles: an experimental study of acceleration*, European Transport Research Review, 6 (2014), pp. 157–170.

[24] ———, *On curve negotiation: From driver support to automation*, IEEE Transactions on Intelligent Transportation Systems, 16 (2015), pp. 2082–2093.

[25] A. Britzelmeier and A. Dreves, *A decomposition algorithm for Nash equilibria in intersection management*, Optimization, (2020), pp. 1–38.

[26] R. Brooks, *A robust layered control system for a mobile robot*, IEEE journal on robotics and automation, 2 (1986), pp. 14–23.

[27] E. F. Camacho and C. B. Alba, *Model predictive control*, Springer science & business media, 2013.

[28] T. Carlson and Y. Demiris, *Collaborative control for a robotic wheelchair: evaluation of performance, attention, and workload*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 42 (2012), pp. 876–888.

[29] L. Chen and C. Englund, *Cooperative intersection management: A survey*, IEEE Transactions on Intelligent Transportation Systems, 17 (2016), pp. 570–586.

[30] P. Cisek, *Cortical mechanisms of action selection: the affordance competition hypothesis*, Philosophical Transactions of the Royal Society B: Biological Sciences, 362 (2007), pp. 1585–1599.

[31] N. S. Council, *motor vehicle safety issues*, 2020.

[32] M. Da Lio, F. Biral, E. Bertolazzi, M. Galvani, P. Bosetti, D. Windridge, A. Saroldi, and F. Tango, *Artificial co-drivers as a universal enabling technology for future intelligent vehicles and transportation systems*, IEEE Transactions on intelligent transportation systems, 16 (2015), pp. 244–263.

[33] M. Da Lio, R. Donà, G. P. R. Papini, and K. Gurney, *Agent architecture for adaptive behaviors in autonomous driving*, IEEE Access, 8 (2020), pp. 154906–154923.

[34] M. Da Lio, A. Mazzalai, and M. Darin, *Cooperative intersection support system based on mirroring mechanisms enacted by bio-inspired layered control architecture*, IEEE Transactions on Intelligent Transportation Systems, 19 (2017), pp. 1415–1429.

[35] M. Da Lio, A. Mazzalai, K. Gurney, and A. Saroldi, *Biologically guided driver modeling: The stop behavior of human car drivers*, IEEE Transactions on Intelligent Transportation Systems, 19 (2017), pp. 2454–2469.

[36] L. De Pascali, F. Biral, and S. Onori, *Aging-aware optimal energy management control for a parallel hybrid vehicle based on electrochemical-degradation dynamics*, IEEE Transactions on Vehicular Technology, 69 (2020), pp. 10868–10878.

[37] J. Decety and J. Grèzes, *The power of simulation: Imagining one's own and other's behavior*, Brain research, 1079 (2006), pp. 4–14.

[38] Y. Demiris, *Prediction of intent in robotics and multi-agent systems*, Cognitive processing, 8 (2007), pp. 151–158.

[39] Y. DEMIRIS AND B. KHADHOURI, *Content-based control of goal-directed attention during human action perception*, Interaction Studies, 9 (2008), pp. 353–376.

[40] Y. DEMIRIS AND A. MELTZOFF, *The robot in the crib: A developmental analysis of imitation skills in infants and robots*, Infant and Child Development: An International Journal of Research and Practice, 17 (2008), pp. 43–53.

[41] M. DIEHL AND S. GROS, *Numerical optimal control*, Optimization in Engineering Center (OPTEC), (2011).

[42] A. DOSHI AND M. M. TRIVEDI, *Tactical driver behavior prediction and intent inference: A review*, in 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2011, pp. 1892–1897.

[43] J. ERDMANN AND D. KRAJZEWICZ, *SUMO's Road Intersection Model*, in Simulation of Urban Mobility, M. Behrisch, D. Krajzewicz, and M. Weber, eds., vol. 8594, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 3–17.

[44] J. E. ESPINOSA, S. A. VELASTÍN, AND J. W. BRANCH, *Detection of motorcycles in urban traffic using video analysis: A review*, IEEE Transactions on Intelligent Transportation Systems, 22 (2021), pp. 6115–6130.

[45] ETSC, *Preventing road accidents and injuries for the safety of employees*, 2011.

[46] ETSI, *Vehicular communications*, standard, 2019.

[47] S. A. FAYAZI AND A. VAHIDI, *Mixed-Integer Linear Programming for Optimal Scheduling of Autonomous Vehicle Intersection Crossing*, IEEE Transactions on Intelligent Vehicles, 3 (2018), pp. 287–299.

[48] T. FLASH AND B. HOCHNER, *Motor primitives in vertebrates and invertebrates*, Current opinion in neurobiology, 15 (2005), pp. 660–666.

[49] T. FLASH, Y. MEIROVITCH, AND A. BARLIYA, *Models of human movement: Trajectory planning and inverse kinematics studies*, Robotics and Autonomous Systems, 61 (2013), pp. 330–339.

[50] J. FLEMING, X. YAN, C. ALLISON, N. STANTON, AND R. LOT, *Real-time predictive eco-driving assistance considering road geometry and long-range radar measurements*, IET Intelligent Transport Systems, (2021).

[51] C. E. GARCIA, D. M. PRETT, AND M. MORARI, *Model predictive control: Theory and practice—a survey*, Automatica, 25 (1989), pp. 335–348.

[52] M. GEMOU, A. SPILIOTIS, AND E. BEKIARIS, *Needs, restrictions and priorities for the development and installation of a multifunctional c-its solution on existing road pavement*, Transportation Research Procedia, 36 (2018), pp. 185–192.

[53] J. GIBSON, *The theory of affordances The Ecological Approach to Visual Perception (pp. 127-143)*, Boston: Houghton Miffin, 1979.

[54] M. GKEMOU, F. BIRAL, I. GKRAGKOPOULOS, G. VALENTI, I. TSETSINAS, E. BEKIARIS, AND A. STECCANELLA, *Cooperation between roads and vehicles: Field validation of a novel infrastructure-based solution for all road users' safety*, Safety, 7 (2021).

[55] R. GRUSH, *The emulation theory of representation: Motor control, imagery, and perception*, The Behavioral and brain sciences, 27 (2004), pp. 377–96; discussion 396.

[56] F. GUAYANTE, A. DÍAZ-RAMÍREZ, AND P. MEJÍA-ALVAREZ, *Detection of vulnerable road users in smart cities*, in 2014 eighth international conference on next generation mobile apps, services and technologies, IEEE, 2014, pp. 307–312.

[57] K. GURNEY, T. J. PRESCOTT, AND P. REDGRAVE, *A computational model of action selection in the basal ganglia. i. a new functional anatomy*, Biological cybernetics, 84 (2001), pp. 401–410.

[58] L. Gurobi Optimization, *Gurobi optimizer reference manual*, 2020.

[59] S. Haendeler, A. Lewandowski, and C. Wietfeld, *Passive detection of wrong way drivers on motorways based on low power wireless communications*, in 2014 IEEE 79th Vehicular Technology Conference (VTC Spring), 2014, pp. 1–5.

[60] C. M. Harris, *Biomimetics of human movement: Functional or aesthetic?*, Bioinspiration & biomimetics, 4 (2009), p. 033001.

[61] C. M. Harris and D. M. Wolpert, *Signal-dependent noise determines motor planning*, Nature, 394 (1998), pp. 780–784.

[62] M. Haruno, D. M. Wolpert, and M. Kawato, *Mosaic model for sensorimotor learning and control*, Neural computation, 13 (2001), pp. 2201–2220.

[63] G. Hesslow, *Hesslow, g. conscious thought as simulation of behaviour and perception. trends cogn. sci. 6, 242-247*, Trends in cognitive sciences, 6 (2002), pp. 242–247.

[64] ———, *The current status of the simulation theory of cognition*, Brain Research, 1428 (2012), pp. 71–79.

[65] T. Hofman, M. Steinbuch, R. Van Druten, and A. Serrarens, *Rule-based energy management strategies for hybrid vehicles*, International Journal of Electric and Hybrid Vehicles, 1 (2007), pp. 71–94.

[66] R. Hult, M. Zanon, G. Frison, S. Gros, and P. Falcone, *Experimental validation of a semi-distributed sequential quadratic programming method for optimal coordination of automated vehicles at intersections*, Optimal Control Applications and Methods, 41 (2020), pp. 1068–1096.

[67] R. Hult, M. Zanon, S. Gras, and P. Falcone, *An miqp-based heuristic for optimal coordination of vehicles at intersections*, in 2018 IEEE Conference on Decision and Control (CDC), IEEE, 2018.

[68] R. Hult, M. Zanon, S. Gros, and P. Falcone, *Optimal Coordination of Automated Vehicles at Intersections with Turns*, in 2019 18th European Control Conference (ECC), IEEE, June 2019.

[69] S. Hurley, *The shared circuits model (SCM): How control, mirroring, and simulation can enable imitation, deliberation, and mindreading*, Behavioral and Brain Sciences, 31 (2008).

[70] *Recommended practice: Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles*, standard, Society of Automotive Engineers (SAE), 2019.

[71] A. Jahn, K. David, and S. Engel, *5g / lte based protection of vulnerable road users: Detection of crossing a curb*, 2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall), (2015), pp. 1–5.

[72] A. M. Jain and N. Tiwari, *Airborne vehicle detection with wrong-way drivers based on optical flow*, in 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015, pp. 1–4.

[73] M. Jeannerod, *Neural simulation of action: A unifying mechanism for motor cognition*, NeuroImage, 14 (2001), pp. S103–9.

[74] M. Jeannerod and V. Frak, *Mental imaging of motor activity in humans. curr opin neurobiol 9:735-739*, Current opinion in neurobiology, 9 (1999), pp. 735–9.

[75] X. Jin, A. Vora, V. Hoshing, T. Saha, G. Shaver, R. E. García, O. Wasynczuk, and S. Varigonda, *Physically-based reduced-order capacity loss model for graphite anodes in li-ion battery cells*, Journal of Power Sources, 342 (2017), pp. 750–761.

[76] L. Johannesson, M. Asbogard, and B. Egardt, *Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming*, IEEE Transactions on Intelligent Transportation Systems, 8 (2007), pp. 71–83.

[77] M. Khayatian, M. Mehrabian, E. Andert, R. Dedinsky, S. Choudhary, Y. Lou, and A. Shirvastava, *A Survey on Intersection Management of Connected Autonomous Vehicles*, ACM Transactions on Cyber-Physical Systems, 4 (2020), pp. 1–27.

[78] N. Kim and A. P. Rousseau, *Comparison between rule-based and instantaneous optimization for a single-mode, power-split hev*, tech. rep., SAE Technical Paper, 2011.

[79] J. Li, X. Wu, M. Xu, and Y. Liu, *A real-time optimization energy management of range extended electric vehicles for battery lifetime and energy consumption*, Journal of Power Sources, 498 (2021), p. 229939.

[80] B. Liu and A. El Kamel, *V2x-based decentralized cooperative adaptive cruise control in the vicinity of intersections*, IEEE Transactions on Intelligent Transportation Systems, 17 (2016), pp. 644–658.

[81] J. Liu and H. Peng, *Modeling and control of a power-split hybrid vehicle*, IEEE Transactions on Control Systems Technology, 16 (2008), pp. 1242–1251.

[82] X. Liu, D. Qin, and S. Wang, *Minimum energy management strategy of equivalent fuel consumption of hybrid electric vehicle based on improved global optimization equivalent factor*, energies, 12 (2019), p. 2076.

[83] R. Lot and F. Biral, *A curvilinear abscissa approach for the lap time optimization of racing vehicles*, IFAC Proceedings Volumes, 47 (2014), pp. 7559–7565.

[84] A. A. Malikopoulos, C. G. Cassandras, and Y. J. Zhang, *A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections*, Automatica, 93 (2018), pp. 244–256.

[85] A. Mazzalai, F. Biral, M. Da Lio, M. Darin, and L. D'Orazio, *Automated crossing of intersections controlled by traffic lights*, in 2015 IEEE 18th International Conference on Intelligent Transportation Systems, 2015, pp. 1928–1933.

[86] A. N. Meltzoff, *The 'like me'framework for recognizing and becoming an intentional agent*, Acta psychologica, 124 (2007), pp. 26–43.

[87] R. C. Miall, *Connecting mirror neuron and forward models*, Neuroreport, 14 (2004), pp. 2135–7.

[88] M. H. Mohamad Nor and T. Namerikawa, *Optimal Coordination and Control of Connected and Automated Vehicles at Intersections via Mixed Integer Linear Programming*, SICE Journal of Control, Measurement, and System Integration, 12 (2019), pp. 215–222.

[89] S. J. Moura, H. K. Fathy, D. S. Callaway, and J. L. Stein, *A stochastic optimal control approach for power management in plug-in hybrid electric vehicles*, IEEE Transactions on Control Systems Technology, 19 (2011), pp. 545–555.

[90] S. J. Moura, J. L. Stein, and H. K. Fathy, *Battery-health conscious power management in plug-in hybrid electric vehicles via electrochemical modeling and stochastic control*, IEEE Transactions on Control Systems Technology, 21 (2013), pp. 679–694.

[91] MQTT.org, *Message queuing telemetry transport*. Accessed: 2021-11-11.

[92] C. Musardo, G. Rizzoni, Y. Guezennec, and B. Staccia, *A-ecms: An adaptive algorithm for hybrid electric vehicle energy management*, European Journal of Control, 11 (2005), pp. 509–524.

[93] A. J. Nagengast, D. A. Braun, and D. M. Wolpert, *Optimal control predicts human performance on objects with internal degrees of freedom*, PLoS Comput Biol, 5 (2009), p. e1000419.

[94] D. S. Naidu, *Optimal control systems*, CRC press, 2002.

[95] E. Namazi, J. Li, and C. Lu, *Intelligent Intersection Management Systems Considering Autonomous Vehicles: A Systematic Literature Review*, IEEE Access, 7 (2019), pp. 91946–91965.

[96] E. R. S. Observatory, *Annual accident report*, 2018. Accessed: 2021-11-11.

[97] S. Onori, L. Serrao, and G. Rizzoni, *Hybrid Electric Vehicles, Energy Management Strategies*, 2016.

[98] W. H. Organization, *Road traffic injuries*. Accessed: 2021-11-11.

[99] G. Paganelli, G. Ercole, A. Brahma, Y. Guezennec, and G. Rizzoni, *General supervisory control policy for the energy optimization of charge-sustaining hybrid electric vehicles*, JSAE review, 22 (2001), pp. 511–518.

[100] G. P. R. Papini, G. Valenti, A. Plebe, and M. Da Lio, *Msprt action selection model for bio-inspired autonomous driving*.

[101] L. V. Pérez, G. R. Bossio, D. Moitre, and G. O. García, *Optimization of power management in an hybrid electric vehicle using dynamic programming*, Mathematics and Computers in Simulation, 73 (2006), pp. 244–254.

[102] J. Rios-Torres and A. A. Malikopoulos, *A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps*, IEEE Transactions on Intelligent Transportation Systems, 18 (2017), pp. 1066–1077.

[103] G. Ripaccioli, D. Bernardini, S. Di Cairano, A. Bemporad, and I. Kolmanovsky, *A stochastic model predictive control approach for series hybrid electric vehicle power management*, in Proceedings of the 2010 American Control Conference, 2010, pp. 5844–5849.

[104] SAE, *Dedicated short range communications (dsrc) message set dictionary*, standard, 2020.

[105] SAFER-LC, *Safer-lc safer level crossing by integrating and optimizing road-rail infrastructure management and design*, 2020.

[106] L. Serrao, S. Onori, and G. Rizzoni, *A comparative analysis of energy management strategies for hybrid electric vehicles*, Journal of Dynamic Systems, Measurement, and Control, 133 (2011), p. 031012.

[107] J. Shin and M. Sunwoo, *Vehicle speed prediction using a markov chain with speed constraints*, IEEE Transactions on Intelligent Transportation Systems, 20 (2019), pp. 3201–3211.

[108] K. A. Smith, *Electrochemical modeling, estimation and control of lithium ion batteries*, (2006).

[109] ——, *Electrochemical control of lithium-ion batteries applications of control*, IEEE Control Systems Magazine, 30 (2010), pp. 18–25.

[110] C. Sun, X. Hu, S. Moura, and F. Sun, *Velocity predictors for predictive energy management in hybrid electric vehicles*, Control Systems Technology, IEEE Transactions on, 23 (2015), pp. 1197–1204.

[111] T. R. Tanim, C. D. Rahn, and C.-Y. Wang, *A temperature dependent, single particle, lithium ion cell model including electrolyte diffusion*, Journal of Dynamic Systems, Measurement, and Control, 137 (2015), p. 011005.

[112] TNO, *Drive c2x deliverable d11.6. final report*, 2014.

[113] E. Todorov and M. I. Jordan, *A minimal intervention principle for coordinated movement*, Advances in neural information processing systems, (2003), pp. 27–34.

[114] J. P. Torreglosa, P. Garcia-Triviño, D. Vera, and D. A. López-García, *Analyzing the improvements of energy management systems for hybrid electric vehicles using a systematic literature review: How far are these controls from rule-based controls used in commercial vehicles?*, Applied Sciences, 10 (2020), p. 8744.

[115] D.-D. Tran, M. Vafaeipour, M. El Baghdadi, R. Barrero, J. Van Mierlo, and O. Hegazy, *Thorough state-of-the-art analysis of electric and hybrid vehicle powertrains: Topologies and integrated energy management strategies*, Renewable and Sustainable Energy Reviews, 119 (2020), p. 109596.

[116] G. Valenti, L. De Pascali, and F. Biral, *Estimation of longitudinal speed profile of car drivers via bio-inspired mirroring mechanism*, in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 2140–2147.

[117] G. Valenti, D. Piscini, and F. Biral, *A cooperative intersection support application enabled by safe strip technology both for c-its equipped, non-equipped and autonomous vehicles*, 06 2019.

[118] F. J. Valero-Cuevas, M. Venkadesan, and E. Todorov, *Structured variability of muscle activations supports the minimal intervention principle of motor control*, Journal of neurophysiology, 102 (2009), pp. 59–68.

[119] L. O. Valøen and J. N. Reimers, *Transport properties of lipf$_6$-based li-ion battery electrolytes*, Journal of The Electrochemical Society, 152 (2005), pp. A882–A891.

[120] N. Vinayaga-Sureshkanth, A. Maiti, M. Jadliwala, K. Crager, J. He, and H. Rathore, *Towards a practical pedestrian distraction detection framework using wearables*, in 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2018, pp. 239–245.

[121] P. Viviani and T. Flash, *Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning.*, Journal of Experimental Psychology: Human Perception and Performance, 21 (1995), p. 32.

[122] A. Wächter and L. T. Biegler, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Mathematical programming, 106 (2006), pp. 25–57.

[123] R. Wang and S. M. Lukic, *Dynamic programming technique in hybrid electric vehicle optimization*, in 2012 IEEE international electric vehicle conference, IEEE, 2012, pp. 1–8.

[124] M. J. Weinstein, M. A. Patterson, and A. V. Rao, *Utilizing the algorithmic differentiation package adigator for solving optimal control problems using direct collocation*, in AIAA Guidance, Navigation, and Control Conference, 2015, p. 1085.

[125] W. L. Winston and J. B. Goldberg, *Operations research: applications and algorithms*, Thomson/Brooks/Cole, Belmont, CA, 2004.

[126] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, *Principles of sensorimotor learning*, Nature Reviews Neuroscience, 12 (2011), pp. 739–751.

[127] D. M. Wolpert, K. Doya, and M. Kawato, *A unifying computational framework for motor control and social interaction*, Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences, 358 (2003), pp. 593–602.

[128] D. M. Wolpert and M. Kawato, *Multiple paired forward and inverse models for motor control*, Neural networks, 11 (1998), pp. 1317–1329.

[129] J. Wu, Y. Zou, X. Zhang, T. Liu, Z. Kong, and D. He, *An online correction predictive ems for a hybrid electric tracked vehicle based on dynamic programming and reinforcement learning*, IEEE Access, 7 (2019), pp. 98252–98266.

[130] S. Xie, X. Hu, S. Qi, X. Tang, K. Lang, Z. Xin, and J. Brighton, *Model predictive energy management for plug-in hybrid electric vehicles considering optimal battery depth of discharge*, Energy, 173 (2019), pp. 667–678.

[131] J. Xing, *Characteristics of wrong-way driving on motorways in japan*, IET Intelligent Transport Systems, 9 (2015), pp. 3–11.

[132] N. Xu, Y. Kong, L. Chu, H. Ju, Z. Yang, Z. Xu, and Z. Xu, *Towards a smarter energy management system for hybrid vehicles: A comprehensive review of control strategies*, Applied Sciences, 9 (2019), p. 2026.

[133] N. Yabuuchi, Y. Makimura, and T. Ohzuku, *Solid-state chemistry and electrochemistry of lico1/ 3ni1/ 3mn1/ 3o2 for advanced lithium-ion batteries iii. rechargeable capacity and cycleability*, Journal of The Electrochemical Society, 154 (2007), pp. A314–A321.

[134] C. Zhai, F. Luo, and Y. Liu, *A novel predictive energy management strategy for electric vehicles based on velocity prediction*, IEEE Transactions on Vehicular Technology, 69 (2020), pp. 12559–12569.

[135] F. Zhang, X. Hu, R. Langari, L. Wang, Y. Cui, and H. Pang, *Adaptive energy management in automated hybrid electric vehicles with flexible torque request*, Energy, 214 (2021), p. 118873.

[136] Y. Zhang, L. Chu, Y. Ding, N. Xu, C. Guo, Z. Fu, L. Xu, X. Tang, and Y. Liu, *A hierarchical energy management strategy based on model predictive control for plug-in hybrid electric vehicles*, IEEE Access, 7 (2019), pp. 81612–81629.

[137] B. Zhou, J. B. Burl, and A. Rezaei, *Equivalent consumption minimization strategy with consideration of battery aging for parallel hybrid electric vehicles*, IEEE Access, 8 (2020), pp. 204770–204781.

[138] I. H. Zohdy and H. A. Rakha, *Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept*, Journal of Intelligent Transportation Systems, 20 (2016), pp. 17–32.