POLITECNICO DI TORINO

Master's Degree in Automotive Engineering

Master's Degree Thesis

Decision Making and Path Planning algorithms for an Autonomous overtaking Vehicle on public road scenario

Supervisors

Prof. Andrea TONOLI

Prof. Nicola AMATI

Eng. Eugenio TRAMACERE

Candidate

Alberto PONSO

October 2022

Abstract

This Master of Science thesis is presenting the study for the transposition of the Path Planning ideas first tackled during Squadra Corse Driverless experience into a real world application: as the number of available Advanced Driver Assistance Systems grows higher, even on medium-end cars, the horizon of a Level 3 SAE Automation level is becoming increasingly closer. This thesis aims at defining a possible Decision Making strategy for a Level 3 automated vehicle to decide how to behave in a realistic extraurban road, prioritizing the Safety above all, and strongly evaluating whether the conditions for overtaking exist in a continuous manner.
The thesis covers the aspects of Decision Making, Path Planning for overtaking as well as the high level controls that are needed to enact the decisions taken; a short section is also dedicated to a brief explanation of the sensing configuration to be employed on the vehicle in order to be able to perform the Decision Making.

Acknowledgements

Other than the obvious thanks to my family, I want to thank in no particular order "the boys", Riccardo Ninfa, Edoardo Branda, Alessandro d'Amico and Giorgio Rolle. Without them, I would still probably be in my BSc feeling lonely and in search of a group of people to do my projects with.
The other big group of people I have to thank is of course Squadra Corse Driverless and all its members, even the ones I never had the chance to meet, with special thanks to Emanuele, Matteo, Antonio, Davide, Stefan, who contributed in making me feel at home.
Umberto Eco, in his "Come scrivere una tesi di laurea" [1] stated that thanking any supervisor is distasteful because *"they only did their job"*; anyway, I still want to thank Eng. Eugenio Tramacere for how he almost babysat me throughout my research and writing of this thesis, as well as Dr. Salvatore Circosta for how he did the same during my BSc thesis.
I should also thank all the mentors one by one, but to reflect the help I got from them, I would risk making the acknowledgements section much longer than the actual thesis, which would be indeed rather unusual.

"ANYTHING IS POSSIBLEEEEEEE"
Kevin Garnett

"Don't believe everything you read on the Internet"
Marcus Tullius Cicero, on his first Reddit post, 63 BC

Table of Contents

List of Tables

List of Figures

Acronyms

ACC
Adaptive Cruise Control

ADAS
Advanced Driver Assistance Systems

AV
Automated Vehicle

CC
Cruise Control

DOF
Degree Of Freedom

DST
Dynamic Single Track

DSTP
Dynamic Single Track with Pacejka Tyre Model

EM
Electric Machine

EV
Electric Vehicle

FSM
Finite State Machine

FoV

Field of View

GNSS

Global Navigation Satellite System

GPS

Global Positioning System

IMU

Inertial Measurement Unit

INS

Inertial Navigation System

LKA

Lane Keeping Assist

LUT

Look-Up Table

MPC

Model Predictive Control

OECD

Organization for Economic Co-operation and Development

SAE

Society of Automotive Engineers

SotA

State of the Art

Chapter 1

Introduction

While the environment in which this thesis work was started - Formula SAE competitions - does not have races wheel by wheel between its participants [2][3], its natural outlet is the application on the road vehicles, as it has always been with vehicle racing [4][5][6]. A focal point of the Squadra Corse Driverless - which constituted its main difference with respect with the "traditional" Squadra Corse Electric - was the presence of a Perception and Path Planning Division[7]: this work of thesis represents the translation of the author's experience into an everyday use for civilian traffic application.
As the team's aim is to build a fully autonomous racing car and to develop its software[7][8], the step to the development of the software side for a road-going vehicle so that it is able to autonomously decide whether it needs and it can overtake a slower leading vehicle was quite natural to be taken; this argument was tackled not only because of its correlation with the Path Planning concept, but also because it was a challenging and relatively novel theme, which can deliver great safety improvements to the road traffic, as overtaking is among the riskiest tasks to be taken during travelling.

1.1 Problem assessment

During this work, we concentrated our focus on an extra-urban road environment, where the overtaking maneuver is often made difficult or impossible because of the presence of vehicle on the oncoming lane: even though the first SAE level 3[9] Automated Vehicles are appearing in highway environment[10][11], we put ourselves into the rural road environment because it is where the overtaking is most difficult and dangerous, due to the presence of oncoming vehicles. The reason for the use of rural roads and highways as Scenarios lies, other than its highest

simplicity compared to urban roads, also in the higher danger intrinsically linked to such roads: as we can see in Table 1.1, the fact that the percentage of **fatal** accidents on motorways is double the percentage of all accidents [**empty citation**] is a sign of the fact that motorways are more dangerous than urban roads This

Department for Transport statistics

Reported Road Casualties Great Britain Annual Report 2020

RAS10001

Reported accidents by speed limit, road class and severity, Great Britain, 2020

Number/percentage change compared to 2019

Type of road	Fatal		Serious (unadjusted)		Serious (adjusted)		Slight (unadjusted)		Slight (adjusted)		All accidents		Road traffic[1]
	Number	% change	Number	% change	Number	% change	Number	% change	Number	% change	Number	% change	% change
Motorway	76	-24	467	-28	515	-31	2.038	-40	1.990	-43	2.581	-38	-25
20 mph[2]	53	-4	1.707	-8	1.795	-11	9.422	-4	9.334	-6	11.182	-5	..
30 mph	468	-14	9.837	-23	10.875	-24	41.899	-25	40.861	-29	52.204	-25	..
40 mph	132	-28	1.718	-17	1.896	-18	5.956	-23	5.778	-27	7.806	-22	..
All built-up roads[6]	653	-16	13.262	-20	14.567	-22	57.277	-22	55.972	-26	71.192	-22	..
50 mph	107	-4	804	-14	894	-15	2.705	-20	2.615	-24	3.616	-18	..
60 mph	473	-14	3.238	-19	3.570	-22	7.604	-23	7.272	-29	11.315	-21	..
70 mph	82	-30	581	-22	633	-24	1.820	-33	1.768	-37	2.483	-30	..
All non built-up roads[6]	662	-15	4.623	-19	5.097	-21	12.129	-24	11.655	-29	17.414	-22	..
Major roads[3]	812	-20	8.389	-22	9.173	-24	33.984	-24	33.200	-28	43.185	-24	-24
Minor roads[4]	579	-11	9.966	-18	11.008	-20	37.469	-22	36.427	-26	48.014	-21	-16
All roads[5]	1.391	-16	18.355	-20	20.181	-22	71.453	-23	69.627	-27	91.199	-22	-21

Table 1.1: Table recapping the shares of accidents and deaths in Great Britain, from [12]

percentage of fatal accidents over total accidents gets even worse if we take into account also the rural roads and not just the motorways: in light of a percentage of 47.35% of total accidents taing place on major roads, 58.38% of the fatal accidents took place there. This unbalance in percentages is due mainly to:

- Lack of pedestrians on major roads.
- Lack of intersections on major roads.
- (For motorways) Lack of Oncoming traffic.

Leading to less accidents on Major roads and

- Higher total speeds.
- Larger differences of speeds.
- Longer trips, leading to more tired drivers.

increasing the fatality and the danger of the accidents on Rural Roads and Motorways/Highways.
Moreover, a study about the road accidents in the state of Maine[13] highlighted that more than two thirds of road accidents occur on rural roads, consistent with the percentage across the whole pool of OECD (Organization for Economic Co-operation and Development, OCSE for the Italian speakers) countries, around 60%.
Moreover, [13] found out that even though head-on crashes - occurring when one vehicle invades the oncoming lane - account for only 5% of the total crashes in the study pool, they were responsible of almost half of the fatalities.
Other than the straight fact that driving on the oncoming lane exposes to the risk of a head-on accident, the overtaking maneuver is imposing a high mental workload on the driver[14], making him more prone to committing mistakes and crashing by himself in case of a long travel. In addition to this, the actions of overtaking and of changing the lane are indicated as the cause of around 5% of the accidents in a study about the distribution of crashes[15], where a correlation between the drivers' age and the accidents caused by improper lane change of said driver was found, as seen in Table 1.2

	Responsible			Non-responsible		
	Aged 16–34 [%]	Aged 35–54 [%]	Aged 55+ [%]	Aged 16–34 [%]	Aged 35–54 [%]	Aged 55+ [%]
Driver not in control	16.4	12.7	8.1	8.5	10.1	9.4
Fail to yield right of way	16.0	15.0	28.9	26.3	21.9	21.5
Misjudge stopping distance	12.0	11.2	8.4	13.2	14.4	14.2
Following too close	9.0	8.7	5.5	10.6	11.1	10.4
Unseen object/person/vehicle	8.1	11.0	13.4	9.9	9.1	10.1
Fail to heed sign/signal	5.1	4.9	7.1	6.0	7.0	8.6
Avoid object/person/vehicle	4.6	4.7	2.2	1.3	4.5	1.3
Improper driving environment	4.3	3.6	1.6	1.3	1.6	1.5
Over speed limit	3.4	1.4	0.3	0.8	0.7	0.8
Improper lane change	2.3	2.9	4.6	4.3	4.1	3.9
DUI	2.2	4.9	1.8	1.9	1.7	1.3
Driver condition	1.8	1.6	1.7	0.4	0.6	0.6
Improper turn/U-turn	1.6	1.9	3.0	2.5	2.5	2.2
Improper passing	1.5	1.4	1.1	1.7	1.8	2.1
Improper backing	1.3	2.0	2.6	1.7	1.9	2.1
Other	10.4	12.1	9.7	9.6	7.0	10.0

Table 1.2: Table recapping the causes of accidents from [15]

Even though such accidents account to less than 5%, - following the approach of [16], who discounted the failure to yield way as the less problematic, yet the most frequent cause of accident - we should not underestimate the risks present in the overtaking maneuver, which is a quite rare occurrence during travel, yet causes a noticeable amount of accidents.
In order to further stress out the magnitude of the problem, we can point out that over 1 million people die every year in car accidents, with over 50 million people

injured[17][18]. This means that a "tiny" 5% can mean, on average, 50 thousand deaths a year and 2.5 million injuries.

WHO REGION	REPORTED DATA[a]		MODELLED DATA[a]	
	n	RATE PER 100 000 POPULATION	*n*	RATE PER 100 000 POPULATION
AFRICAN REGION	52 302	7.2	234 768	32.2
REGION OF THE AMERICAS	139 466	15.5	142 252	15.8
SOUTH-EAST ASIA REGION	143 977	8.4	285 020	16.6
EASTERN MEDITERRANEAN REGION	76 912	14.1	175 668	32.2
EUROPEAN REGION	113 346	12.8	117 997	13.4
WESTERN PACIFIC REGION	135 316	7.6	278 321	15.6
GLOBAL	**661 319**	**10.1**	**1 234 026**	**18.8**

[a] Adjusted for 30-day definition.

Table 1.3: 2008 Road accident deaths from [17]

1.2 The SAE levels of automation

In order to tackle the matter of driver lack of attention (as well as exhaustion due to long drives) and therefore to reduce the accidents, Driver Assistance Systems were introduced, although very primitive, already in the 1950s [19] with the introduction of the Cruise Control; however, it wasn't until the mid 1980s that electronic CC came to the series market in full swing[20]. The second main family of Driver Assistance Systems is represented by the Lane Keeping Assist (LKA), which however came much later [20] because of their need for more complicated perception systems[21].

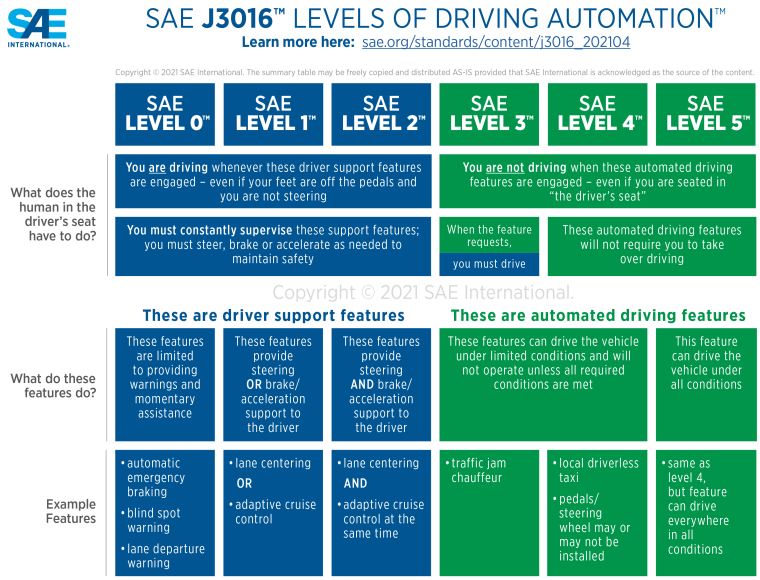

Figure 1.1: The Automation levels as defined by SAE in [9]

The aforementioned CC (later replaced by its evolution ACC) and LKA are the backbone of the definition of SAE levels of automation as per [9], which can be seen in Figure 1.1: if only one of the two systems is present or both are present but for whatever reason they can not work together, the car is classified as SAE level 1; on the contrary, if both LKA and ACC can be enacted together, the car can be classified as SAE level 2, but the driver is still legally responsible for everything happening during the operation of the vehicle.
As of the time of writing, only Mercedes-Benz has obtained the Level 3 certification[11], which offloads the human driver of legal responsibility while the car is driving itself, for its Drive Pilot system, making the first step towards the so-called Highway Chauffeur[22]. It is to be noted, however, that even this project is still far from a "real" application: the car is limited to a speed of 60 km/h (this limit makes it classified as "Traffic Jam Chauffeur", rather than "Highway Chauffeur"[23]), can only be operated autonomously during the day and with low levels of moisture on the road and only on the Autobahn, with controlled access and no intersections. Because of this, we can affirm that the technology is still in its first days, therefore it should be no surprise that this work of thesis is still unrefined.
In our idea, the SAE level associated with our vehicle should be at least 3, as the design of the system was carried out to activate both the LKA and the ACC together continuously and the cases in which the driver would be asked to take back the control of the vehicle should be very limited, making the model on the threshold of a Level 4 vehicle. Because our car is able to perform lane changes and

overtakes, while running at speeds well higher than 60 km/h, we will refer to the Automated Driving function offered by our vehicle as "Highway Chauffeur"; it is to be noted, however, that the scenario on which we developed it is the much more challenging rural road environment, with oncoming vehicles and intersections.

1.3 State of the Art

1.3.1 State of the Art: Perception and Localization

In order to ensure that the car is able to travel autonomously, we need to be able to give information to the car about where it is placed. The process of Localization can be conducted in an **absolute** manner or in a **relative** manner, both of which have their strengths and weaknesses and require different hardware to be installed[24].

Absolute Localization

The process of Absolute Localization aims at finding the position of the vehicle in a global set of coordinates, generally the GPS coordinates set; Global Navigation Satellite System (GNSS) is the most common localization system on the market, with a share of 13% of all the 200 millions traveling in Europe[25], and is now almost guaranteed on all new production cars. Despite their penetration in the market and the maturity of the technology, which is about to enter its 50th year of existence[26], yet the GPS alone is not enough for a real autonomous car, as its precision can be in the range of the tens of meters[27][28]; this precision level is satisfying for the navigation of a manned vehicle, where the low level control is performed by the human driver and the GNSS is only giving high level indication ("in 300 meters, turn left"), but this value is absolutely inadequate for the lateral localization of an unmanned vehicle, where even an error of just 1 meter could lead to a mistake in the lane placement and - most severely - it could endanger the safety of pedestrian and cyclists on the road[29].
Another possibility to localize the vehicle in an absolute reference system is the Inertial Navigation System (INS), which is based on the double integration of the the accelerations that the Inertial Measurement Unit (IMU) undergoes, in order to know the position change with respect to the starting position; the latter, however, has to be input to the INS in some way[30], like for example through GNSS itself. What was pointed out as the main weakness of the GNSS (the low precision), however, is present with the INS as well, since the signals of the IMU are quite noisy.

Relative Localization

A completely different approach, however, consists of localizing the road features like the lane boundaries and the other road users into a reference frame which is centered on the car itself[31][32].
This is the approach that was followed during this work of thesis, as computing the global positions of the lane boundaries resulted too computationally heavy for the computer used to run the simulations as the Matlab Simulink block which performs lane recognition from the Driving Scenario outputs the results in said local reference frame.

Our Approach

Even though the work of thesis was only simulated on Matlab Simulink and did not reach such a readiness level as to be put into practical experimentation, yet the working of the Simulink block performing lane recognition has an output which is mimicking the one of a real Lane Detection method[33].
The goal in general is to obtain the lateral offset of the two lane boundaries[34] as well as their orientation in order to get the lateral distance and orientation error of the car with respect to the centerline which is to be followed, by mean of a Stanley controller[35].

1.3.2 The Algorithms for the Overtaking

Since this is a thesis on the Path Planning for Autonomous Overtaking, the main part of the State of the Art section is going to be devoted to the State of the Art of the Path Planning in Autonomous Overtaking and - more in general - in the Obstacle Avoidance (of which we can consider Autonomous Overtaking a subcase). The different methods which were evaluated are:

- Ellipses of Influence [36]: this method deals with not only Autonomous Overtaking, but with the whole matter of Collision Avoidance.
- Polynomial Curves: these can be like [37], where the polynomial is used to describe the local errors or like [38][39], where instead the polynomial function describes the planned path itself.
- Driving Corridors [40][41] approach consists of the minimization of a cost function in global coordinates with the prediction of the obstacles' movement inside a feasibility corridor.

- State Lattices [42] are a way in which all the possible movements of the vehicles are previously discretized in primitive movements and then a lattice grid of all the possible resulting paths are represented.
- Sigmoid Curves [43][44][45]: with this approach, a path resembling a sigmoid function is drawn and then the vehicle is required to follow the path.

Below, we will give a quick explanation of the strengths and weaknesses of each approach, compared to our needs and in the end highlight which one we chose and why. A more complete and exhaustive review on the subject can be found in [46][47].

Ellipses of Influence

This method, proposed by [36], attempts to obtain a safe navigation by a mobile robot in a populated environment, avoiding collisions with the obstacles which are there placed. The choice of an ellipse, the generic shape for a circle, is justified by the desire to "have a more generic and flexible mean to surround and fit accurately different kind of obstacles shapes" [36], as can be easily seen in Figure 1.2.

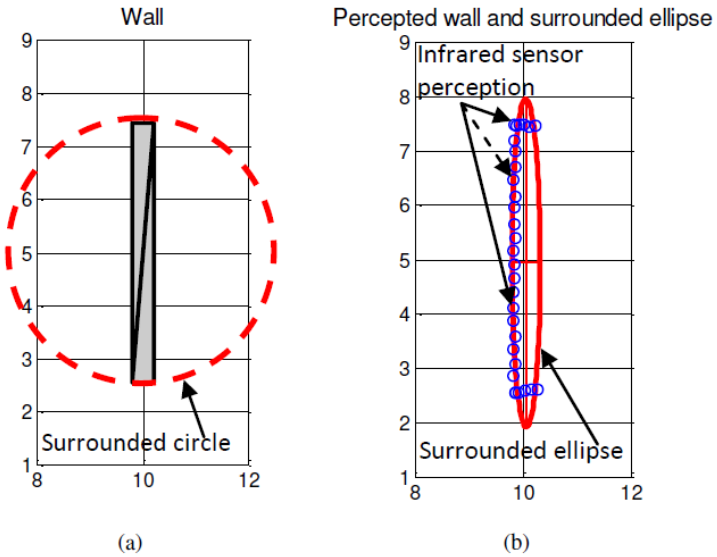

Figure 1.2: The comparison between circle (a) and ellipse (b) enveloping of obstacles. From [36]

The solution with circles (a) yields a much bigger forbidden area compared to the one with ellipses (b), which is much more representative of how much space the obstacle is taking; the way the obstacle is defined is coherent with our proposed hardware, as the use of a RADAR would yield a series of points, which are then

used to define the ellipse as shown once again in 1.2 (b). However, this is a weakness too, as the radar of the Ego Vehicle will most likely only detect points belonging to the rear end of the vehicle, therefore the car would be perceived as an obstacle only for a smaller portion than what it actually is; of course, the car would detect other points when being side by side, but in our view this is undesirable.
Another reason which led us to discard this solution is the fact that the controls are then selected among 4 different possibilities:

- Clockwise target attraction.
- Counter-clockwise target attraction.
- Clockwise obstacle avoidance.
- Counter-clockwise obstacle avoidance.

Apart from the fact that counter-clockwise would not be of course a possibility, as overtaking on the right is extremely dangerous and prohibited, the definition of two different sets of controls would be a bit pointless in our opinion as well as computationally heavy. The shortcomings that we have pointed out are due to the fact that this is not an algorithm designed specifically for automotive applications, but rather for a holonomic robot in a crowded environment (as could for example be a warehouse), therefore the counter-clockwise evasion maneuver is feasible, as well as the fact that objects can be placed obliquous to the trajectory of the vehicle, an occurrence which is rather unusual for a road vehicle. Nevertheless, it was an interesting approach and we still decided to include it in the State of the Art section, especially considering that most unmanned vehicle applications come from this sector.

Polynomial Curves

The first polynomial curves approach that we are going to consider is the one followed by [37],in which the polynomial function is used to define the trend of the pose error $e = [e_x \ e_y \ e_\theta]^T$.
In [37], the definition of the aforementioned pose error is then reduced to $e = [e_x \ e_y]^T$ and it is found by solving the equation 1.1

$$\begin{aligned} e_x^d(t) &= a_{0x} + a_{1x}(t - t_0) + a_{2x}(t - t_0)^2 + a_{3x}(t - t_0)^3 \\ e_y^d(t) &= a_{0y} + a_{1y}(t - t_0) + a_{2y}(t - t_0)^2 + a_{3y}(t - t_0)^3 \end{aligned} \tag{1.1}$$

with the boundary conditions 1.2 defined with the goal of achieving maximum smoothness in the transition between the previous LKA scenario and the current

Lane Change scenario

$$\begin{array}{rclcrcl} e_x^d(t_0) & = & e_{x0} & & e_y^d(t_0) & = & e_{y0} \\ \dot{e}_x^d(t_0) & = & \dot{e}_{y0} & ; & \dot{e}_y^d(t_0) & = & \dot{e}_{y0} \end{array} \tag{1.2}$$

and the other four boundary conditions 1.3 defined to obtain a stable nil error at the end of the lane change maneuver.

$$\begin{array}{rclcrcl} e_x^d(t_f) & = & 0 & & e_y^d(t_f) & = & \Delta v_{Rx}(t_f) \\ \dot{e}_x^d(t_f) & = & 0 & ; & \dot{e}_y^d(t_f) & = & 0 \end{array} \tag{1.3}$$

for a total of eight equation for eight unknowns (a_1 ... a_8).

Opposite to this approach by [37], [38][39] computes directly $[x, y]$.
On one hand, [38] defines the trajectory through a pair of equations where we need to find 11 coefficients to optimize (equation 1.4).

$$\begin{aligned} p_x &= g_{0x} + g_{1x}t + g_{2x}t^2 + g_{3x}t^3 + g_{4x}t^4 \\ p_y &= g_{0y} + g_{1y}t + g_{2y}t^2 + g_{3y}t^3 + g_{4y}t^4 + g_{5y}t^5 \end{aligned} \tag{1.4}$$

On the other hand, [39] solves the path planning by using three different approaches:

- Approach I: used for the LKA, where the longitudinal coordinate x is described through a fourth order equation, while the lateral coordinate y is described through a function of x, as seen in equation 1.5 in order to define the trajectory starting from the path

$$\begin{aligned} x &= a_1t^4 + a_2t^3 + a_3t^2 + a_4t + a_5 \\ y &= f(x) = f(a_1t^4 + a_2t^3 + a_3t^2 + a_4t + a_5) \end{aligned} \tag{1.5}$$

  the boundary conditions, which are then imposed, lead to solve the matricial equation 1.6 which give out the coefficients $q = [a_1 \ ... \ a_5]^T$

$$A \cdot q = B \tag{1.6}$$

  where the A matrix depends on the initial time t_0 and the final time interval t_f as of equation 1.7, while the B matrix depends on the initial x coordinate x_0 and the "final" position x_e, belonging to the local point on the path from equation 1.8:

$$A = \begin{bmatrix} t_0^4 & t_0^3 & t_0^2 & t_0 & 1 \\ t_f^4 & t_f^3 & t_f^2 & t_f & 1 \end{bmatrix} \tag{1.7}$$

$$B = \begin{bmatrix} x_0 \\ x_e \end{bmatrix} \tag{1.8}$$

- Approach II: is used in the overtaking/lane change scenario, where the y coordinate is not defined anymore as a function of x, but as a polynomial function with different coefficients, as is pointed out in equation 1.9:

$$\begin{aligned} x &= a_1t^4 + a_2t^3 + a_3t^2 + a_4t + a_5 \\ y &= b_1t^7 + b_2t^6 + b_3t^5 + b_4t^4 + b_5t^3 + b_6t^2 + b_7t + b_8 \end{aligned} \tag{1.9}$$

  as it can be easily seen, it requires to find 13 different coefficients; the definition of the boundary conditions is beyond the scope of this work and, therefore, we redirect to the original paper [39]. The scope we had in presenting such equations was to highlight the large complexity of the polynomial formulation to solve this problem.

- Approach III: is used in order to keep the safety distance from the leading vehicle and it complies with parts of the A approach.

The reason for such a wide tractation of this type of solution is simple: among the approaches presented in the SotA section, this is one of those which most influenced us and, therefore, we felt like we needed to dedicate a large portion of the work of thesis to introduce it.

Driving Corridors

This approach, presented by [40][41], has similarities between the two presentations, which both start from an *a priori* defined map, where the Ego Vehicle is localized by mean of camera recognition [40]. Both approaches then have a part concerning with the prediction of the movements of all the other vehicles around, together with their occupancy areae and related uncertainty, which allows to recognise if the corridors are suitable or not.
The reason why we refused to use such method from the very beginning is linked to both the need for an *a priori* map and, most importantly, the need for a prediction on the trajectories of all the vehicles around, which would prove too heavy for our hardware.

State Lattices

This approach to path planning was first introduced by Pivtoraiko, Knepper and Kelly in [48] to discretize the search space, in order to reduce the computational cost, by avoiding to search the whole continuum.
The term *lattice* is used to define a vectorial space which represents the space discretization applied by [48], as can be seen in Figure 1.3.

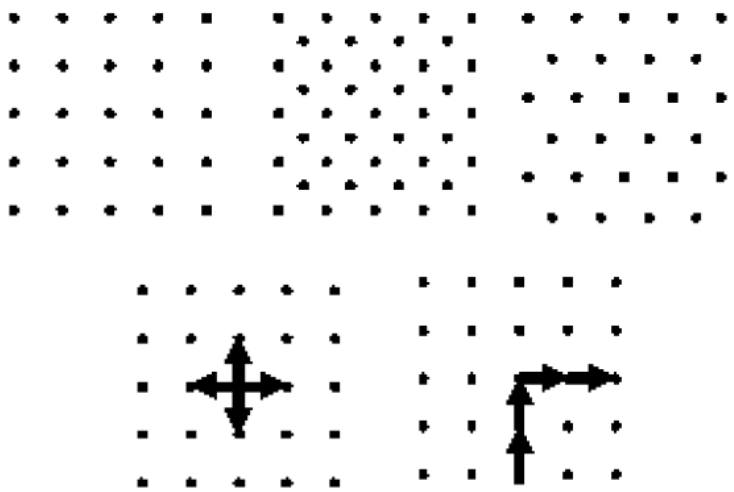

Figure 1.3: "Regular Lattices. Top: rectangular, diamond, and triangular (hexagonal) lattices. Bottom Left: Controls for a 4-connected lattice. Bottom Right: Discontinuous heading change", from Figure 1 of [48]

The robot can then move between these points with a series of movements which are predetermined, like driving straight or turning with a constant radius; even in the case of a single possible turn radius chosen (Figure 1.4), it is easy to see how after just three movements in series the path space is already quite dense and - most importantly - the resulting path is something which looks very natural for a car trajectory, as we can see from Figure 1.5.
The way this type of path planning is then integrated with the Autonomous Overtaking task is similar to the way it is performed in the Driving Corridors approach: the best path is selected through a cost function and all the possible paths to be weighted are those descending from the lattices matrix.
The reason why we discarded this method is therefore similar to the one which led us to discard the Driving Corridors one: in order to compute a cost function we would have to generate several different paths and then evaluate them all in order to compute their cost, therefore multiplying the computational cost of the Path Generation. The lattices method was developed to reduce the computational cost, but this solution did not appeal to us, because the cost reduction was achieved through the constraint on the possible maneuvers, by discretizing the steering angle in few possible values; we wanted to leave all the possible steering angle values available to our vehicle and, most importantly, we wanted a function which planned a Path and then computed the steering angle to follow such Path, similarly to the solution adopted in the Squadra COrse DRIVERLESS experience.

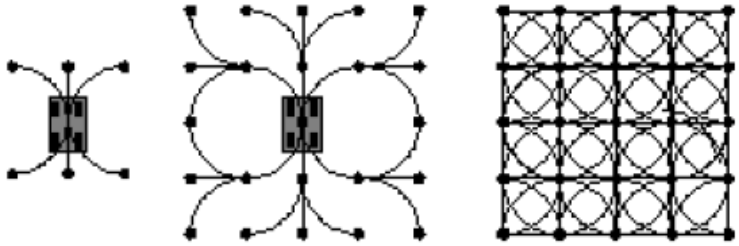

Figure 1.4: Image of the expansion of a lattice matrix with respectively 1 maneuver (left), 2 (center) and 4 (right). From Figure 2 of [48]

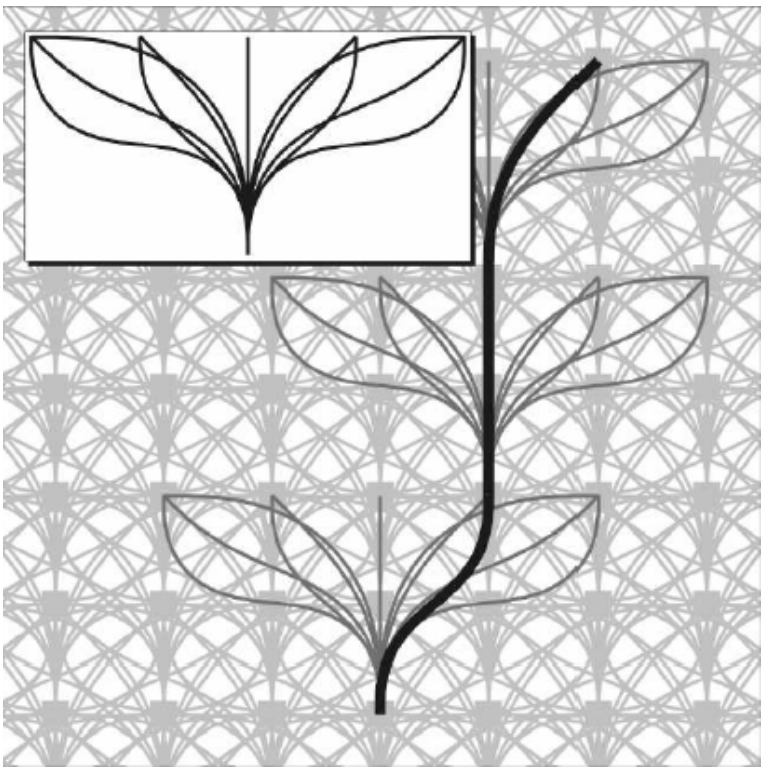

Figure 1.5: In top left: a single position with all the primitive maneuvers included in its particular model. Main image: the resulting lattice space; in thick black, the selected path. From Figure 3 of [48]

Sigmoid Curves

The last approach to the Path Planning for the Overtaking is represented by the use of the Sigmoid Curves; this method, unlike all the ones previously reported, is

intrinsically connected to the concept of Lane Change. In fact, differently from the other State of the Art methods, the Sigmoid Curves see application only in case of overtaking, to define the path to follow for the Lane Change maneuver [43][44][45]; the reason for this peculiarity can be easily found by the definition of the Sigmoid function in equation 1.10

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1.10}$$

which yields a profile as the one shown in Figure 1.6. The name sigmoid, in fact, comes from the S-shaped curve, ranging from 0 to 1;

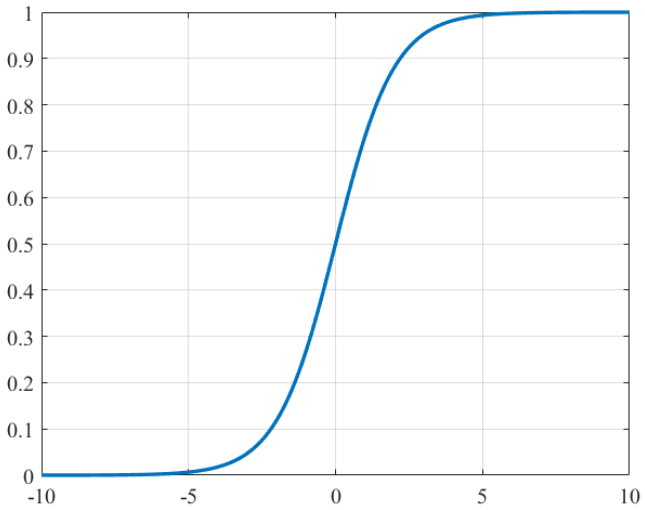

Figure 1.6: A Sigmoid Curve

that particular shape, with a steep increase around the 0 mark is the reason why this function is so used in sectors like Artificial Intelligence [49][50][51], to represent a binary function *"GO - NO GO"* by using a continuous variable, trying to mimick the firing of the neurons [52][53]; on the other hand, such sharp and rapid movement between two parallel lines can be a very good approximation of an overtake or a Lane Change maneuver, where the car wants to stay in its own lane for as long as possible, in order to both exploit as much as possible the draft effect as well minimize the risk of frontal crash with oncoming vehicles [54][55].
The approach used by [43] requires to draw a path to be followed which has a fixed length, named **Horizon length**, which is recomputed once the vehicle has followed part of the trajectory, for a length of **Horizon step length**, in a rolling horizon fashion like the one of the Model Predictive Controller (MPC) [56][57]; however, because of the heavy computational cost involved, we did not consider this part

and only focused on the mathematical formulation for the path generation in case an obstacle is detected.
In order to obtain the path $[X_r\ Y_r]$, a circular curve is designed around the obstacle, which is located in $[X_0\ Y_0]$ of radius S (S being the lateral distance we want to have while overtaking, because of Safety reasons), which is defined as a vector of length N as of equation 1.11

$$Y_r(N) = \sqrt{S^2 - (X(N) - X_0)^2} + Y_0 \tag{1.11}$$

The main weakness of this formula, which makes it unusable for driving purposes lies in the fact that the passage from a straight line to the circular part makes the path not derivable, and therefore impossible to be followed, as shown in Figure 1.8.

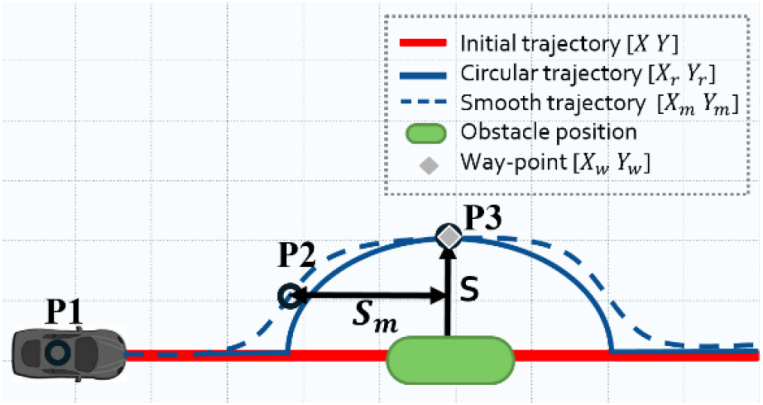

Figure 1.7: Figure 6 of [43], showing the need for a smoother curve

In order to obtain a feasible path from this otherwise useless geometry, a smoother profile is obtained, shown in Figure 1.8 as a dashed line: as can be seen, it is obtained as the composition of two sigmoids, one for the lane change on the left and one to come back to the rightmost lane; the equation governing such shape is 1.12

$$Y_m(N) = \frac{Y_w}{(1 + e^{C(d(N) - S_m)})} \tag{1.12}$$

where Y_w is the width of the movement that the car must perform, to change the lane, C is a factor influencing the slope of the sigmoid (the higher C, the sharper is the sigmoid, as of Figure 1.8), d is the distance from the leading vehicle and S_m is a safety distance we should keep from the leading vehicle.

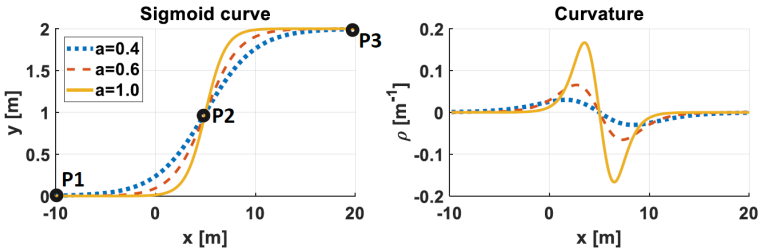

Figure 1.8: Figure 2 of [43], giving a sensitivity analysis on the value of C, here called a

Overall speaking, the solutions for the sigmoid presented by [44] and [45] are the same as the one presented above, differing only for the conventions regarding the names of the variables and the way the path is then followed, so we will assume to have covered the whole class of solutions by presenting just [43].

1.3.3 Conclusions about the Algorithms for Overtaking

After having presented in a brief way the main options for the Overtaking Algorithms, with their strengths and weaknesses, we want to conclude the section by pointing out what was our decision; because of all was said above, the choice ultimately fell on the Sigmoid Curves: the reason for doing so, lies in the absolute freedom of movement, without a series of primitive maneuvers (which led us to the discard of the *State Lattices* together with their computational cost) and the relatively easy function, differently from the *Polynomial Curves*, which requires to be solving a system with many Equations and Unknowns; moreover, the presence of a coefficient in the exponential part of the Sigmoid (C in equation 1.12) allows for an adaptive slope of the Sigmoid Curve as a function of Speed, as will be pointed out in Section 2.1.

1.4 Thesis Outline

The thesis will follow this structure: in **Section 2** we will present the methodology we have followed during the work, from the approach we had to the models we used (including their constituting equations); in **Section 3** we will present and comment the results obtained during the experimental testing; finally, in **Section 4**, we will take a look at the whole project and the results we obtained, to sum up the whole work and we will expose some potential future developments for this work.

Chapter 2

Methodology

This chapter is devoted to the exposition of the Methodology we followed during this work of thesis: a first look will be given to the actual process of thought and the bottom-up construction we have adopted during the tests of the decision making functions and the Sigmoid function; after this, we will devote a section to the complete vehicle model we have introduced for the extensive testing. Later on, the algorithms used for the lane recognition and the Oncoming/Leading vehicle analysis will be shown; a further section will be dedicated to the methodology of the above-mentioned equations for the Path Planning itself, while the last section of this chapter will be used to point out the principles of our validation processes and the step-by-step approach we followed, whose results will be expressed and commented in Section 3.

2.1 Bottom-Up Approach

As already pointed out in the section 1.3.2, the main advantage of the Sigmoid Curves, which ultimately led us to choose them as our Path Planning algorithm is their high customability, which allows for a good tuning: in particular, we decided to tune the K parameter, which defines the slope of the sigmoid curve, as a function of speed.
In order to tune this, we put ourselves in an empty environment, with just a car in front and our "vehicle" as a simple DSTP (Dynamic Single Track with Pacejka Tyre) model, also known as Single Track Model, Monotrack model or Bicycle model [58], which will be described later in section 2.2; the reason for the use of such a simple model lies in the high computational cost of the simulation that we were to run: in order to build up a Look-Up Table of values of K for different velocities, we ran a series of simulations for each constant velocity, with a K value running from 0.011 to 0.061 with a step of 0.0005, for a total of 101 simulations. The metric to

evaluate the best K value for that particular speed was the mean of the tracking error e_y and then the best K value was saved up in a vector.
This K-sweep is extremely time consuming, taking more than 15 minutes on a very performing PC per a single speed value by just using the simpified DSTP model, therefore the reason for its use is quite clear. Moreover, once the K for the overtake Sigmoid Curve was obtained (as seen in Figure 2.1), the same sweep was performed

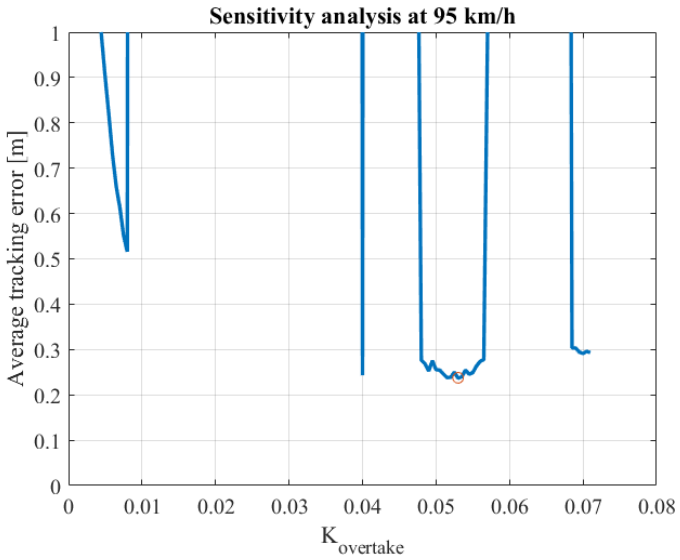

Figure 2.1: First step of the sensitivity analysis on the $K_{overtake}$ value

on the go-back Sigmoud Curve, in order to get the K values for the LUT used for the go-back path (Figure 2.2), which presents different characteristics compared to the one for the overtake, as will be seen in the relative sections 2.9 and 2.10; once the correct K_{goback} value was obtained, the sweep on the first K, the $K_{overtake}$ was repeated: if the best K was not different from the first one found **before** the determination of the ideal K_{goback} (as shown in Figure 2.3, then the two values are saved each in the corresponding LUT, otherwise, the sweep on the K_{goback} is repeated and so on in a trial-and-error fashion until both values converge. Such trial-and-error process was repeated for a wide range of speeds, resulting extremely time-consuming, but in the end providing a good LUT for both the $K_{overtake}$ and the K_{goback} values as a function of the speed, thereby providing a novel approach not yet present in literature, given the fact that throughout the research phase of this work of thesis we found no paper presenting the idea of an adaptable Sigmoid Curve slope as a function of vehicle speed at the moment of the start of overtaking. The resulting values of the LUT are shown in Figure 2.4 and Figure 2.5; one thing which can be noticed is that, for low speeds, the K can be outside of the range previously defined for the sweep: this is due to the fact that, at low speeds, the

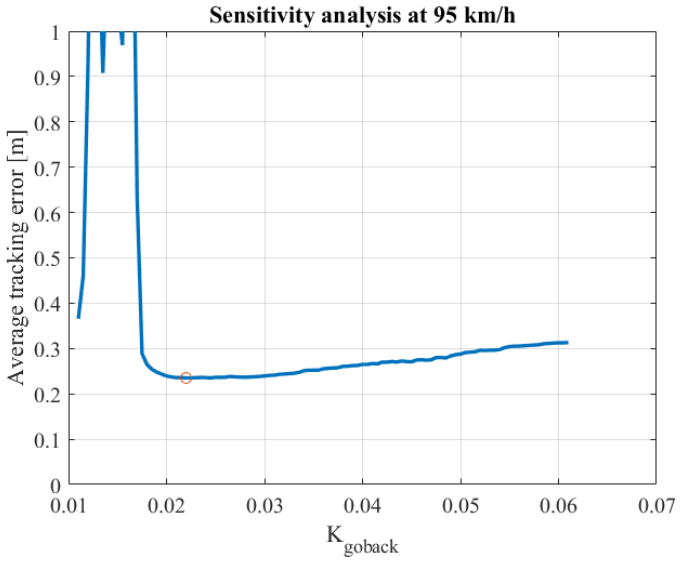

Figure 2.2: First step of the sensitivity analysis on the K_{goback} value

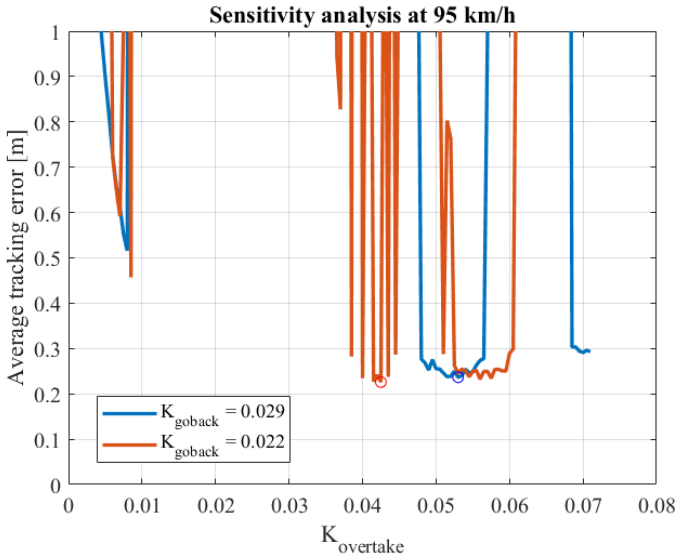

Figure 2.3: Second step of the sensitivity analysis on the $K_{overtake}$ value

trajectory can be much sharper and still ensure a stable behaviour, plus - at low speeds - the car takes a longer time to complete the overtake (the key value is actually the speed difference, but a slower car still takes longer time to run the sigmoid).

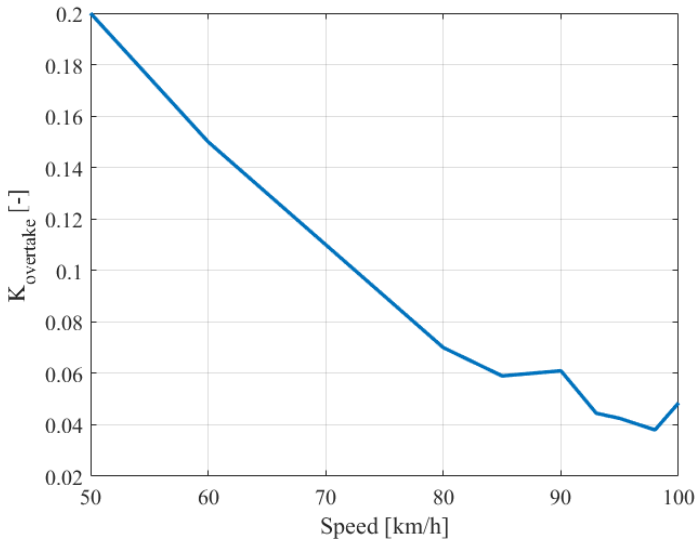

Figure 2.4: Result of the sensitivity analysis on the $K_{overtake}$ value

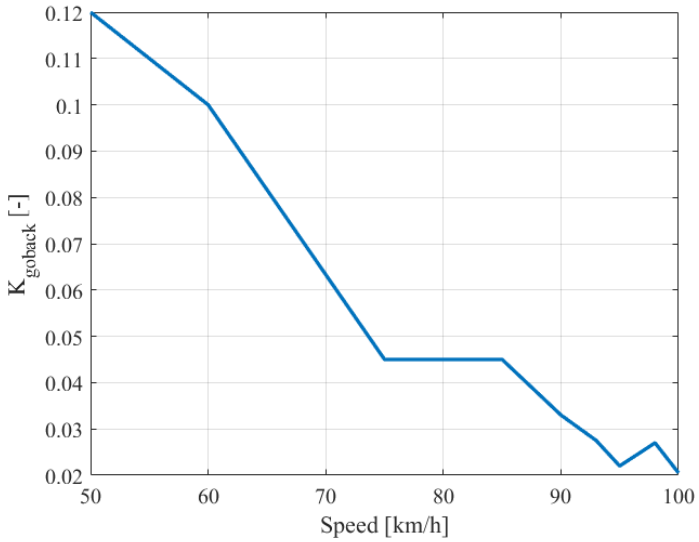

Figure 2.5: Result of the sensitivity analysis on the K_{goback} value

2.2 The bicycle model

As already stated at the beginning of the Section 2.1, the first tests and sensitivity analyses were conducted using a simplified vehicle model in order to speed up the first phase of experimentation. The simplified vehicle model in object is the

so-called Bicycle Model [58] or DST (Dynamic Single Track) which collapses the two wheels of an axle into a single wheel.

The model is schematized into Figure 2.6 and keeps as characteristic geometries

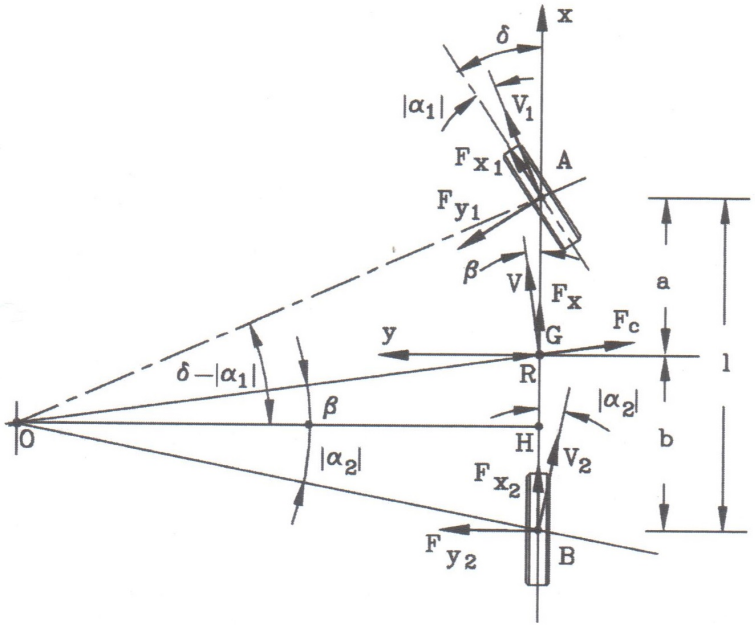

Figure 2.6: DSTP model scheme, from Figure 25.12, page 265 of [58]

of the vehicles only the wheelbase and the longitudinal position of the centre of gravity, while neglecting the front and rear track and the vertical position of the centre of gravity; these are not directly accounted for the kinematic of the vehicle but can still be considered when computing the load transfer due to the lateral acceleration. According to Section 25.3 of [58], in order to use the most simplified single track model, the following assumptions need to be fulfilled:

- The vehicle is moving at constant speed on a constant radius curve.
- The road is level.
- The radius of the path R is much larger than the car wheelbase l.
- Aerodynamic forces are neglected.
- Aligning torques are neglected.

While the assumption on the level road is acceptable and the curvature radius R is generally much larger than the wheelbase l, the assumption on the constant speed and especially the constant radius is not acceptable, and the aerodynamic forces

are difficult to neglect, considering we will operate on rural roads or even highways. Therefore, the model equations we used are following the theory from Section 25.5 of the same reference, which require the following assumptions (integrally quoted from [58]):

- The sideslip angle of the vehicle β and of the wheels α are small. The yaw angular velocity $\dot{\psi}$ can also be considered a small quantity.
- The vehicle can be assumed to be a rigid body moving on a flat surface, i.e. roll and pitch angles are neglected as well as the vertical displacements due to suspensions.

In this way, we managed to achieve the DSTP model without sacrificing too many assumptions, also known as the 3-DOF model because of its 3 degrees of freedom, namely $[x\ y\ \psi]^T$; the strength of this model also lies in the fact that, despite neglecting the pitch and the roll angles as degrees of freedom *per se*, it can account for the longitudinal and lateral load transfer through their influence on the front and rear axle cornering stiffnesses C_1 and C_2.
To start laying out the equations, first of all we point out the existence of two different reference frames:

- The first reference frame is an inertial one, of coordinates $[X\ Y\ Z]^T$ and $[\Phi\ \Theta\ \Psi]^T$ which will be later called "the Global Reference Frame" and is fixed to the ground, with origin in the starting center of the vehicle at time t_0. It will be used to compute the position of the vehicle through integration.
- The second reference frame is a non inertial one, of coordinates $[x\ y\ z]^T$ and $[\phi\ \theta\ \psi]^T$ which is the vehicle reference frame, fixed to its center of gravity.

We can now begin to analitically consider the model, starting with Newton's second law in equation 2.1

$$\begin{cases} m\ddot{X} = F_X \\ m\ddot{Y} = F_Y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.1}$$

where the uppercase letters represent the global reference frame, with the lowercase z due to the fact that the axis of rotation z and Z (yaw angle ψ and Ψ) are parallel and therefore the same.
The forces in the global reference frame are of no interest to us, so we want to move to the $[F_xF_y]^T$ vector of local forces, which can be done through equation 2.2

$$\begin{bmatrix} F_X \\ F_Y \end{bmatrix} = \begin{bmatrix} cos(\psi) & -sin(\psi) \\ sin(\psi) & cos(\psi) \end{bmatrix} \begin{bmatrix} F_x \\ F_y \end{bmatrix} \tag{2.2}$$

local forces are, compared to the global ones, much more useful, as they are the forces resulting from the tires in the longitudinal and lateral directons, as can be obtained from Pacejka magic formula, first defined in 1986 by Professor Pacejka, which can be further elaborated in [59]. In order to link the equations of system 2.1 with the local forces, we need to pass to a system of local accelerations and local forces like 2.3:

$$\begin{cases} m\ddot{x} = F_x \\ m\ddot{y} = F_y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.3}$$

a possible way is through the Lagrangian approach, which goes out of scope of this work of thesis: we refer to [58] for this approach; instead, we will use the speed composition approach, based on the *basic kinematic equation*, from which we can derive the derivative in time performed in the inertial frame of a generic vector located in the local frame:

$$\left.\frac{d\vec{V}}{dt}\right|_{inertial} = \left.\frac{d\vec{V}}{dt}\right|_{local} + \vec{\Omega} \times \vec{V} \tag{2.4}$$

If we take as generic $\vec{V}$ the velocities vector $\vec{v}$, we have the following vectors:

$$\vec{v} = \begin{bmatrix} u \\ v \\ 0 \end{bmatrix} \quad ; \quad \vec{\Omega} = \begin{bmatrix} 0 \\ 0 \\ \psi \end{bmatrix} \tag{2.5}$$

putting together equations 2.3 and 2.5, we end up with the Newton's II Law for the local reference frame

$$\begin{cases} m(\dot{u} - \dot{\psi}v) = F_x \\ m(\dot{v} + \dot{\psi}u) = F_y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.6}$$

which can be easily reconducted to the intuitive $F = m \dots a$ formula on a straight, where the yaw rate is considered nil.
Further on, it is possible to introduce the *side slip angle* β, the angle between the speed vector and the longitudinal axis of the vehicle

$$\beta = atan\left(\frac{v}{u}\right) \tag{2.7}$$

from which we can revert

$$\begin{cases} u = Vcos(\beta) \approx V \\ v = Vsin(\beta) \approx V\beta \end{cases} \tag{2.8}$$

which, going into equation 2.6 yield

$$\begin{cases} m(\dot{Vcos}(\beta) - \dot{\psi}Vsin(\beta)) = F_x \\ m(\dot{Vsin}(\beta) + \dot{\psi}Vcos(\beta)) = F_y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.9}$$

Since the side slip angle β is generally small and seldom goes above 10° [58][60], we can therefore perform first order Taylor expansion on equation

$$\begin{cases} m(\dot{V} - \dot{\psi}V\beta) = F_x \\ m(\dot{V\beta} + \dot{\psi}V) = F_y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.10}$$

However, since the term $\dot{\psi}v$ is much smaller than $\dot{u}$ and the side slip angle β is small, the equations of 2.6 can be simplified directly into

$$\begin{cases} m\dot{V} = F_x \\ m(\dot{v} + \dot{\psi}V) = F_y \\ J_z\ddot{\psi} = M_z \end{cases} \tag{2.11}$$

The system of equations 2.11 can be easily decoupled, provided that we can decouple the longitudinal and lateral forces on the wheels (i.e. we neglect the elliptical model correlation between F_x and F_y); in this way the first equation accounts only for the longitudinal dynamics, while the other two are influencing only the lateral dynamics.
The final equations we used are the ones simplifying the Pacejka Magic Formula [59] - which will be fully used in the Full Model and is described in Section 2.5.4 - in order t reduce the computational cost. The simplified equation is presented in Equation 2.12

$$F_{y_i} = p_1 sin\Big(p_2 atan\Big(p_{3,i}\alpha_i - p_4\big(p_{3,i}\alpha_i - atan(p_{3,i}\alpha_i)\big)\Big)\Big) \tag{2.12}$$

Starting by the definition of side slip angle (equation 2.7) and the geometry of Figure 2.6, it is easy to demonstrate that

$$\begin{cases} \alpha_f = \beta + \dfrac{a}{V}\dot{\psi} - \delta_1 \\ \\ \alpha_r = \beta - \dfrac{b}{V}\dot{\psi} \end{cases} \tag{2.13}$$

2.3 The Experimental Approach

After we have obtained the trend of the K for different values of the speed V with a simplified DSTP model, we moved towards a more realistic application of our

model for decision making and controls: to do so, we have moved towards a more complete Vehicle Model, featuring a real Electric Motor, thus giving limits on the acceleration we can obtain in every instant, and 4 separate wheels with their own slip and dynamics, as is presented in Section 2.5.
Once we have "installed" on the Full Vehicle model the different controls that we want to use and the needed sensors, we will test the controls by running the vehicle in different scenarios and check that the behaviour of our EgoVehicle in said scenario is going according to what we expected, therefore "validating" the controls and the working principles of that particular state of the State Machine (each state is going to be fully detailed in Section 2.11).
The scenarios have been developed with Matlab app **Driving Scenario Designer**, of which an example is shown in Figure 2.7.
Just like we did for Section 2.1, we are going to follow a Bottom-Up approach also

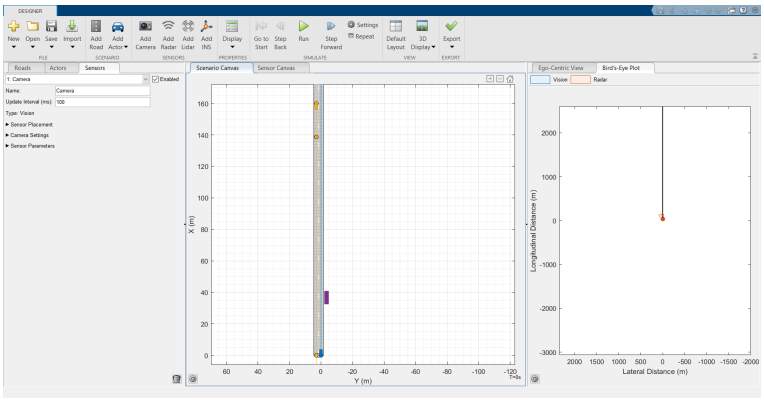

Figure 2.7: A screenshot from the Driving Scenario Designer App

for the experiments on the Full Vehicle model: in fact, the first Driving Scenarios we used are going to be simple Scenarios which can be assessed as Level 1 on the SAE Automation levels (Figure 1.1), as we are going to start by testing the controls **singularly** in order to ensure that the single control is properly designed; a following step will be to try the vehicle on a hybrid Scenario between the first two Scenarios in order to perform a verification of the Level 2 Automatic Driving; of course, when we say "verification" we of course also refer to a very long process of *trial and error* involving the tuning of controllers, sensors and post-processing functions for the sensors which took several hours for every Driving Scenario.
Once the model has been confirmed to work in Level 2 Automated Driving, we moved to a series of Scenarios with both Leading and Oncoming vehicles in progressively more complex environments to try and put our system in difficulty and force errors, so that we could improve on what was wrong.

2.4 The Driving Scenarios

This section will present a short list of the various Scenarios we used to test our system, in a "chronological order": the first Scenarios in the list are going to be the first ones we tested, which - according to Section 2.3 are also going to be the easiest ones. After the shortlist, we will comment on them one by one and give some basic information of them.

- ACC_Scenario, which we used to test the Adaptive Cruise Controller.
- Skidpad, a simple circular road we used to test the Lane Keeping Assist (LKA) in an easy environment, with constant radius cornering.
- Empty Turn Scenario, a rural road with a wide bend and oncoming vehicles.
- Mountain Road, a rural road with challenging turns to test the LKA in a real world environment.
- Mountain Road Circuit, a closed loop version of Mountain Road.
- Overtake Scenario, a simple straight where the EgoVehicle completes an Overtake maneuver.
- Corner Overtake, where the Overtake is performed on a road which is not straight.
- Double Overtake, where the EgoVehicle successfully completes an Overtake maneuver, comes back in its lane and then performs a second one.
- Multiple Overtake, where the second Overtake maneuver involves two vehicles in a column.
- Abort Scenario, where the vehicle is forced to abort its Overtake maneuver.
- Emergency Overtake, where the vehicle performs an Emergency Overtake maneuver.
- Abort Oncoming Vehicle, where the EgoVehicle has to abort the Overtake in order to avoid an oncoming vehicle.
- Abort Brake, where the EgoVehicle has to heavily brake after the Overtake was aborted in order to avoid rear-ending the lead vehicle.

2.4.1 ACC_Scenario

This Scenario was used to test and tune the Adaptive Cuise Controller: to do so, a road with two lanes - one per direction - was created through the Driving Scenario Designer App; the centerline is continuous, so that the overtake is never made possible and the EgoVehicle is forced to follow the LeadVehicle. Such LeadVehicle has a varying speed, in order to put the ACC in a difficult situation and starts with a consistent advantage on us, so that at the beginning we do not have a leading vehicle. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
300	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.6	-3.5	0

We can see from the above tables that the scenario is not empty, but still not too complicated, given that it was used to validate the Adaptive Cruise Control. In order to validate the working of the ACC, the leading vehicle has a variable speed, so that the ACC system is put into a complex situation.
To perform a validation of the ACC, we want to also test the data analysis function

(Section 2.7), which is tasked with the recognition of Leading and Oncoming vehicles: if this function was not working properly, we would see a random Leading vehicle while we encounter the Oncoming vehicle or we would have a very noisy Speed profile, with random peaks due to miscalculations of the Leading vehicle speed.
Another actor which is a constant through many of the Scenarios is the presence of a Parked vehicle on the right side of the road, to ensure that the State Machine works properly and the car does not move into the GO BACK state as soon as it passes a car which is on the right of the Ego Vehicle.

2.4.2 Skidpad

This second Scenario is probably the simplest of all the ones we tested: in fact this is an empty Scenario, without any other vehicle, not even a Parked vehicle on the side; it was used as a very simple test Scenario in order to check that everything was right with the Stanley controller, from the sign of the command to the gain value (Section 2.8.3) and to check the use of the correct units of measurement; it was a very trivial check, but it was important to perform it in order to avoid silly mistakes and nevertheless we wanted to include it in this thesis work to show all the steps we followed in the testing of our controls and our decision making.
Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 1.
- **Double direction road:** NO.
- **Lane Marker:** NO.
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

X [m]	Y [m]	Yaw [°]
0	0	0
300	300	90
0	600	180
-300	300	270
0	0	0

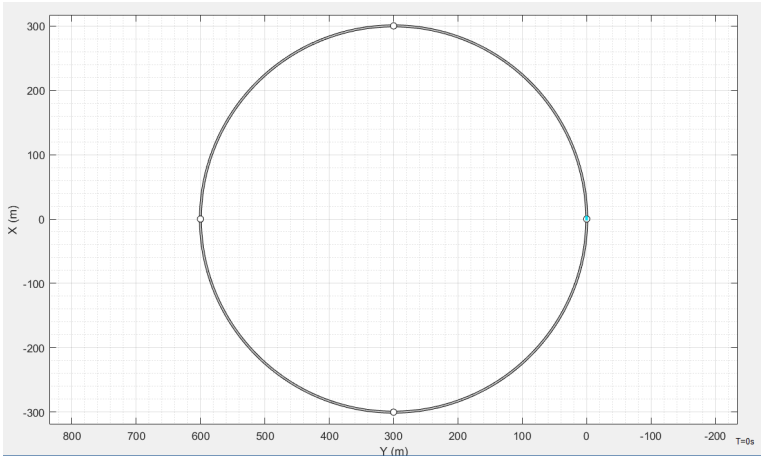

Figure 2.8: The map of the Skidpad track from the Driving Scenario Designer App

2.4.3 Empty Turn Scenario

This third Scenario was used to perform a first easy test on the LKA performed through the Stanley Controller; in order to do so, we drew two straights and linked them with a sharp left turn (Figure 2.9); the goal was to try and check the limits for the whole Perception and Planning pipeline while trying to take said corner at a moderately high speed, over the 80 km/h mark. As with ACC_Scenario, there is also an oncoming vehicle which is used to test the Leading/Oncoming function and - most importantly - to check that the Lane Detection, and therefore the LKA, is working also in case of an oncoming vehicle.
Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

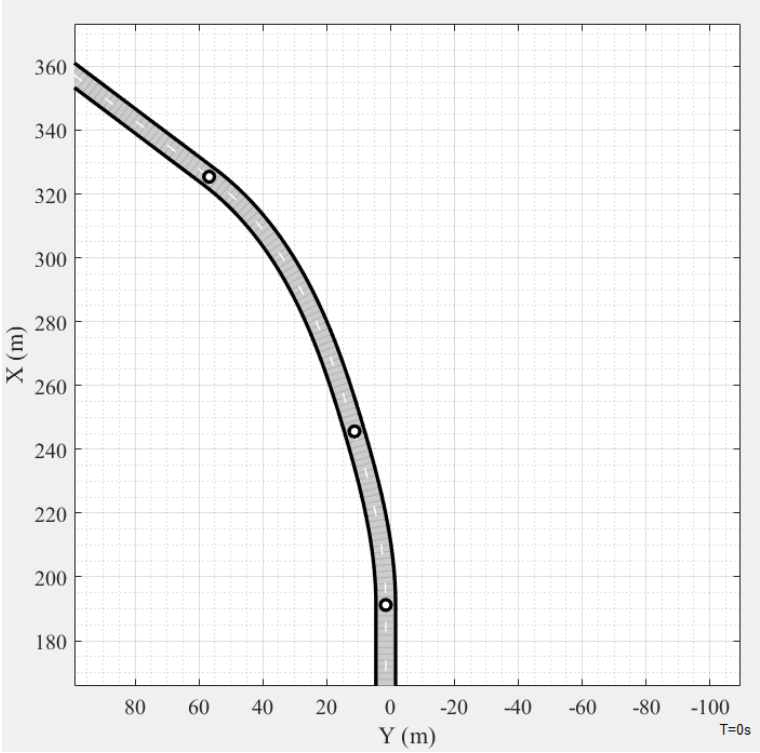

Figure 2.9: The map of the Empty Turn Scenario track from the Driving Scenario Designer App

X [m]	Y [m]	Yaw [°]
0.0	1.5	0
191.2	1.5	0
245.6	11.3	16.35
325.4	56.8	53
385.6	136.7	53
686.5	536.0	53
987.4	935.3	53
1288.3	1334.6	53

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
137.7	3	108.0
115.6	3	108.0
58.0	3	108.0
-300.0	3	108.0

2.4.4 Mountain Road

In order to test the working of the Stanley Controller at low speeds and for small radius corners, we created a fourth scenario (Figure 2.10), characterized by said low speed small radius corners, which was dubbed "Mountain Road" because of its inspiration from winding roads which are characteristic of mountains.
Below are the specifics of this Scenario:

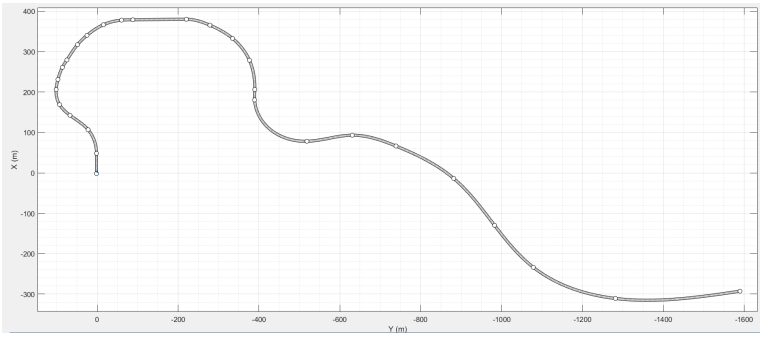

Figure 2.10: The map of the Mountain Road track from the Driving Scenario Designer App

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 50 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

X [m]	Y [m]	Yaw [°]
-1.8	1.8	0
48.2	1.8	0
106.8	22.6	40.44
142.5	67.4	53.35
168.9	93.1	30.61
206.1	101.7	-3.25
231.1.	97.1	-14.95
261.2	86.2	-27.46
279.4	75.0	-33.13
317.3	48.8	-40.07
340.3	25.3	-50.08
366.6	-15.3	-66.07
377.9	-60.2	-84.91
378.8	-88.2	-89.67

X [m]	Y [m]	Yaw [°]
380.1	-220.4	-90.00
365.5	-278.6	-113.30
332.6	-334.9	-129.57
278.7	-376.4	-155.99
206.3	-389.7	178.72
180.5	-389.3	-176.70
78.2	-518.7	-88.60
93.2	-630.8	-89.57
66.4	-738.9	-112.21
-13.8	-881.7	-130.24
-129.8	-982.7	-142.59
-233.8	-1078.7	-128.03
-292.8	-1589.7	-81.60

2.4.5 Mountain Road Circuit

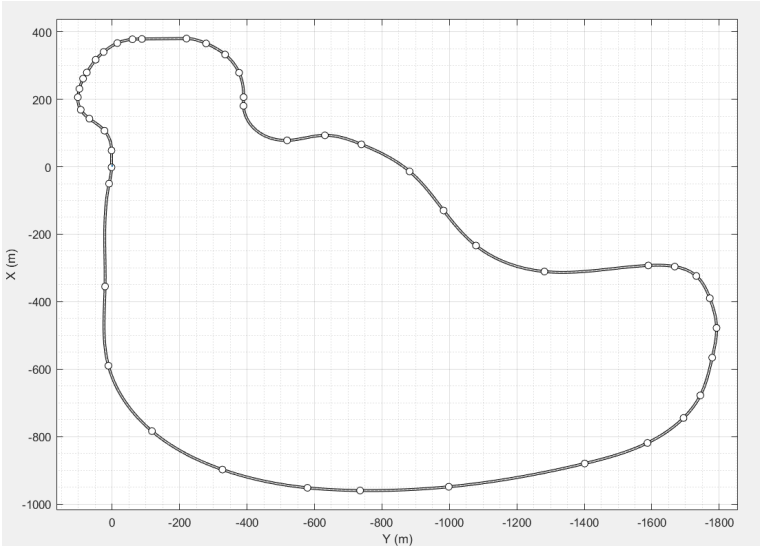

Figure 2.11: The map of the Mountain Road Circuit track from the Driving Scenario Designer App

After the Mountain Road Scenario, we concluded our LKA testing with a closed loop version of that Scenario (Figure 2.11) whose details are listed below:

- **Lane Width:** 3 m (constant).

- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 50 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

X [m]	Y [m]	Yaw [°]
-1.8	1.8	0
48.2	1.8	0
106.8	22.6	40.44
142.5	67.4	53.35
168.9	93.1	30.61
206.1	101.7	-3.25
231.1.	97.1	-14.95
261.2	86.2	-27.46
279.4	75.0	-33.13
317.3	48.8	-40.07
340.3	25.3	-50.08
366.6	-15.3	-66.07
377.9	-60.2	-84.91
378.8	-88.2	-89.67
380.1	-220.4	-90.00
365.5	-278.6	-113.30
332.6	-334.9	-129.57
278.7	-376.4	-155.99
206.3	-389.7	178.72
180.5	-389.3	-176.70
78.2	-518.7	-88.60
93.2	-630.8	-89.57
66.4	-738.9	-112.21

X [m]	Y [m]	Yaw [°]
-13.8	-881.7	-130.24
-129.8	-982.7	-142.59
-233.8	-1078.7	-128.03
-310.8	-1281.7	-95.51
-292.8	-1589.7	-87.20
-296.0	-1668.0	-100.09
-324.0	-1732.0	-131.40
-390.0	-1772.0	-159.30
-478.0	-1792.0	-178.71
-566.0	-1779.0	167.08
-678.0	-1744.0	152.89
-745.0	-1694.0	134.76
-819.0	-1587.0	115.97
-880.0	-1401.0	103.47
-949.0	-998.0	95.45
-960.0	-735.0	89.37
-952.0	-579.0	84.23
-898.0	-327.0	70.82
-784.0	-118.0	49.79
-590.0	11.0	14.85
-355.0	21.0	-1.09
-50.2	9.1	-11.47
-1.8	-1.8	0

2.4.6 Overtake Scenario

After concluding the tests on the LKA system, we moved to finally testing the Autonomous Overtaking system, the core of this thesis work. To do this, we designed several different Scenarios which start from the base of the **ACC_Scenario** of Section 2.4.1, differing between them only for the other actors involved in the

Scenario.
This first Overtaking Scenario was used to perform a simple test of the Sigmoid Creation and the Overtake Maneuver with the Full Vehicle Model and a leading vehicle perceived through the Sensing equipment instead of simply copying the exact value of position from the leading vehicle as we did in Section 2.1.
Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
64.41	-0.15	54.0
75.70	0	54.0
81.80	0.10	54.0
95.30	0.10	54.0
100.10	0.10	54.0
4800.00	-1.50	54.0
5000.00	-1.50	54.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
23.48	-3.88	0

2.4.7 Corner Overtake Scenario

This Scenario was developed in order to have an environment in which the EgoVehicle overtakes the leading vehicle even in a situation in which the road is not perfectly straight; this test was needed in order to demonstrate what we will later state in Section 2.9.2 i.e. that the local Sigmoid Curve definition allows us to perform an Overtake even if the road is not perfectly straight, in counterposition to what was instead true for the global Sigmoid Curve definition that was discussed in Section 2.9.1.
Of course, since the Overtake is a dangerous maneuver *per se* and we want to perform it at a high speed, the Scenario Track can not be as curved as the one

presented in Section 2.4.4 or even Section 2.4.3, therefore we designed a Road which is only slightly curved, but is anyway enough to prove our point; the map can be seen in Figure 2.12
We have run two different simulations onto this Scenario, with two different starting positions of the lead vehicle in order to test two different overtaking spots. Below are the details:

Early Overtake

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

X [m]	Y [m]	Yaw [°]
0	1.5	0
200	1.5	0.70
900	50.0	5.60

- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
63.46	-0.26	36.0
85.45	-0.26	36.0
129.71	-0.36	36.0
171.23	-0.26	36.0
227.00	0.40	36.0
276.30	1.60	36.0
327.10	3.20	36.0
415.00	7.30	36.0
462.80	10.40	36.0
518.50	14.40	36.0
555.20	16.90	36.0
638.30	24.00	36.0
683.70	27.90	36.0
788.10	37.50	36.0

Late Overtake

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **Road Trajectory**

X [m]	Y [m]	Yaw [°]
0	1.5	0
200	1.5	0.70
900	50.0	5.60

- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
397.01	6.54	36.0
462.80	10.40	36.0
518.50	14.40	36.0
553.70	17.10	36.0
638.30	24.00	36.0
683.70	27.90	36.0
787.40	37.60	36.0
810.20	40.00	36.0
867.10	45.20	36.0

2.4.8 Double Overtake

We developed this Scenario, which is not much more complex than the previous Scenarios Overtake Scenario and Corner Overtake, in order to check that, after a first Overtake maneuver, everything was reset and a second Overtake could be performed just like the first one. In order to be sure that the Overtakes take place as we would expect, the centerline marker is dashed and there is only an oncoming vehicle at the very beginning of the Scenario, while the Track is the usual straight road. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.

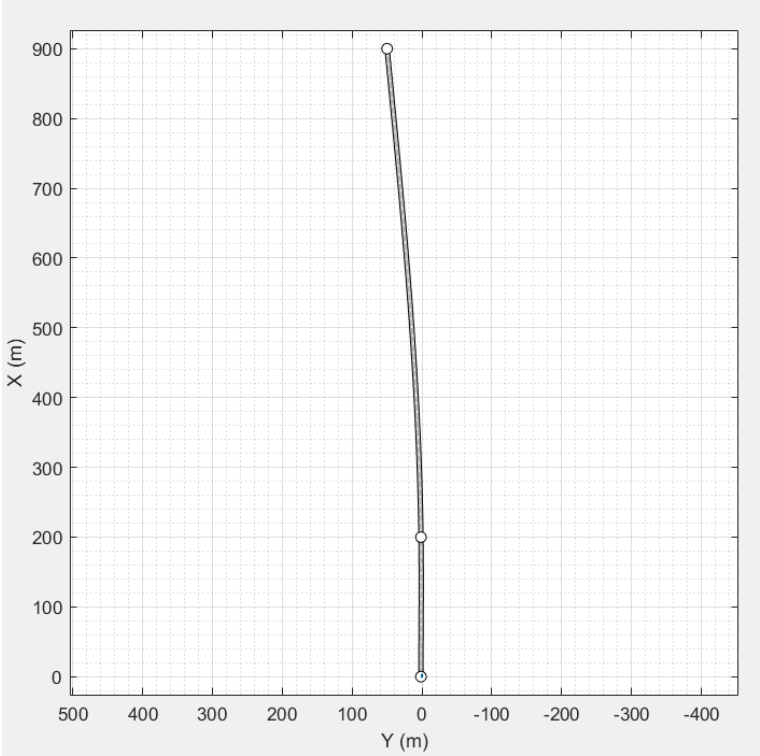

Figure 2.12: Map of the Corner Overtake Scenario

- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **First leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
300	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **Second leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
1000	0	54.0
5000	0	54.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

2.4.9 Multiple Overtake

This Scenario was developed as a further evolution of the Double Overtake Scenario, in order to not only check that the EgoVehicle can perform two separate overtake maneuvers in the same Scenario, but also that in a single maneuver we are able to overtake two vehicles. We can say that this Scenario was more to test our decision making capabilities (Section 2.11) than the Controls themselves. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **First leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
300	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **Second leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
1000	0	54.0
5000	0	54.0

- **Third leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
1050	0	54.0
5000	0	54.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

2.4.10 Abort Scenario

This Scenario was the first of a series of Scenarios we developed in order to test and experiment the Abort process and the stability of the vehicle in such situation as well as of the Decision Making. In the first Scenario, carrying on the bottom-up approach we followed throughout the entire thesis work, we started with a quite easy situation, with no Oncoming vehicles at the moment of the Overtake (just the one at the very beginning of the simulation) and the OvtCounter which gets raised through the Detection of a continuous Lane Marker instead of the previous Dashed Marker. In order to create a Scenario in which the Abort maneuver took place while the EgoVehicle was already partially steering, but not late enough to trigger an Emergency Overtake instead of the Abort, we fiddled with the starting position of the leading vehicle and the percentage of road to be marked with Dashed Marking and the one with Solid Marking. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [16% dashed, 22% solid, 62% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.

- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
235	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

2.4.11 Emergency Overtake

This Scenario has been created by mean of simply receding the position of the lead vehicle compared to where the lead vehicle was starting in the Abort Scenario, so that when the Lane Marker becomes Dashed the EgoVehicle is already too committed in the Overtake maneuver and therefore cannot go into Abort State, but needs to move into Emergency Ovt State. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [16% dashed, 22% solid, 62% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
195	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

We can see that it is heavily inspired by the previous Double Overtake Scenario as is evident from the Actors involved, which are the same with the addition of a Third leading vehicle, located just 50 meters in front of the Second one.

2.4.12 Abort Oncoming Vehicle

This Scenario was created in order to test the working of the *Abort* State: particularly, after having ensured that the EgoVehicle could abort the Overtake (Section 2.4.10) in case we met a Solid Lane Marker, we wanted to ensure that the EgoVehicle could detect a second oncoming vehicle after it had detected one at the beginning of the Scenario and that it could detect it while already being following the lead vehicle; in order to build this Scenario, we placed a Second oncoming vehicle so that it could enter in range of the Side Radar of our EgoVehicle around the same time when - in Abort Scenario (Section 2.4.10) - we encountered the Solid Lane Marker. On the contrary, this Scenario has only the Dashed Lane Marker, so that the only factor raising OvtCounter can be the oncoming vehicle. Below are the specifics of this Scenario:

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [100% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.

- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
195	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **First oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Second oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
1140	4.1	72.0
940	4.1	72.0
600	4.1	72.0

(this Actor is scripted to appear and start the trajectory only at 24.7 s of the simulation)

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

2.4.13 Abort Brake

This Scenario was a sort of a "Final Test" for our Decision Making and Path Planning System: we fiddled with the position of the leading vehicle, so that the transition between Dashed Lane Marker and Solid Lane Marker happened during the Overtake in a way that when the State jumped from 3 (OVERTAKE) to 5 (ABORT), we were on the boundary between the two Lanes; this meant that the Abort maneuver was the most difficult possible maneuver because

- The Overtake maneuver was in its most possible advanced status (if the Ovt-Counter was raised later, we would have moved into State 4 (EMERGENCY OVT).
- The lead vehicle was very close, so the risk of rear-ending it was actually very high.

Below are the specifics of this Scenario

- **Lane Width:** 3 m (constant).
- **Number of Lanes:** 2.
- **Double direction road:** YES.
- **Lane Marker:** [16% dashed, 22% solid, 62% dashed].
- **Starting EgoVehicle speed:** 85 km/h.
- **Starting EgoVehicle acc command:** 0.
- **leading vehicle trajectory**

X [m]	Y [m]	V [km/h]
228	0	72.0
500	0	64.8
1000	0	54.0
5000	0	72.0

- **oncoming vehicle trajectory**

X [m]	Y [m]	V [km/h]
160	3	72.0
139	3	72.0
0	3	72.0
-300	3	72.0

- **Parked Truck**

X [m]	Y [m]	V [km/h]
33.60	-3.50	0

2.5 The Full Vehicle model

We are going here to present the Full Vehicle model which was employed in order to test the Finite State Machine (FSM); because the main focus of this thesis work is not the development of a Full Vehicle model, we are going to discuss just briefly the systems we use, with references to sources to deepen the knowledge on them. Said Full Vehicle model is divided into 3 main blocks:

- Steering.
- Motor and Driveline.
- Vehicle and Terrain.

which we are going to discuss one by one with the addition of the Gear Selector.

2.5.1 Gear Selection System

The Gear Selection System is a very simple algorithm where the Selected Gear Number is a function only of the Vehicle Speed and of the previous Gear Number, as shown by the following PseudoAlgorithm: As said, the Gear Selection Algorithm is

Algorithm 1 Gear Shifting

```
function GN(V, GN(n − 1)) ▷ where GN(n − 1) is the previous GN and V is expressed in m/s
    if GN(n − 1) is 1 then
        if V > 25 then
            GN(n) = 2
        else
            GN(n) = 1
        end if
    else if GN(n − 1) is 2 then
        if V > 40 then
            GN(n) = 3
        else if V < 20 then
            GN(n) = 1
        else
            GN(n) = 2
        end if
    else
        if V>33 then
            GN(n) = 3
        else
            GN(n) = 2
        end if
    end if
    output GN(n)
end function
```

extremely basic, without even accounting for the required Power or considering the Efficiency Map of the Electric Machine or similar. We would also like to highlight that the shift-up speeds do not correspond to the reverse shift-down speed, i.e. the speed at which we shift **down** to I gear from II gear is lower than the speed at which we shift **up** from I gear to II gear; this is due to avoid a "noisy" gear profile, with continuous shift up and down when the EgoVehicle is around the shift speed.

2.5.2 Steering System

The description of the steering System is going to be even briefer than the description of the Gear Selection System: while the Steering System Model from [61] is designed to be able to mount a 4 Wheel Steering System, we decided to use only the Front Axle to steer, with a gain of 1 from the desired Steering Angle coming from the Stanley Controller of Section 2.8.3.

2.5.3 Motor and Driveline

The Power Unit of our EgoVehicle is going to be an Electric Machine, whose parameters are going to be available in the Appendix, which is activated by a command called ***acc command*** which is one of the two Look-Up Table (LUT) arguments, together with the EM rotational speed.
The EM rotational speed is linked to the speed of the EgoVehicle through the rotational speed of the wheels and, actually, is **only** related to the speed of the wheels, rather than the speed of the EgoVehicle: since this is a Full Vehicle model, in fact, it would be incorrect to assume that $V_{veh} = Rw \cdot \omega_w$, since the slip is a factor to be considered; therefore the logical chain of dynamic equations is the following:

- Dynamic Equation of Equilibrium of the 4 wheels.
- (Since the Vehicle is a 2WD Front Wheel Drive) Front Differential.
- Gear Ratio of the Differential.
- Gear Ratio of the selected gear.

To write the equation for the rotational speed of the EM, we consider the above steps and the result is:

$$\omega_{em} = \frac{\omega_{FL} + \omega_{FR}}{2} \tau_f \tau_{gb} \tag{2.14}$$

Because of the smaller Moment of Inertia of the Electric Machine, compared to the one of an Internal Combustion Engine (about half, comparing [62] and [63]), we assumed that the torque output by the EM was equal to T_{em} obtained from the LUT instead of

$$T_{out,em} = T_{em} - J_{em}\dot{\omega}_{em} \tag{2.15}$$

The total torque input to the Front Differential is

$$T_{in,diff} = T_{em}\tau_{gb}\tau f \tag{2.16}$$

which is then split to the two Front Wheels by mean of a Limited Split Differential which transfers a percentage of the Torque proportional to the difference between the two wheel speeds as in Equation 2.17

$$\begin{cases} \Delta T = K \cdot tanh(\omega_{FL} - \omega_{FR}) \\ T_{FL} = \dfrac{T_{diff}}{2} - \Delta T \\ T_{FR} = \dfrac{T_{diff}}{2} + \Delta T \end{cases} \tag{2.17}$$

One important point we want to mention here is that, for the sake of simplicity, we excluded the braking command from the Full Vehicle model, demanding it to the EM negative torque; this was actually one of the main reasons, together with the fastest transient and the fact that Electric Vehicles (EV) penetration is fastly increasing [64], which led us to choose the electric propulsion in opposition to an Internal Combustion Engine.
To represent the braking power, we assumed that, in case we are braking, the rear axle is able to provide the same Torque (negative) as the one which the EM is outputting to the front axle; this is represented by a switch on the value of the State which enables torque (upperbounded to 0) on the Rear wheels only in case we are in States 5 (ABORT), 6 (GO BACK) or 7 (BRAKE), as shown by Figure 2.13.

Of course, this solution would not be feasible in real life and is actually not a

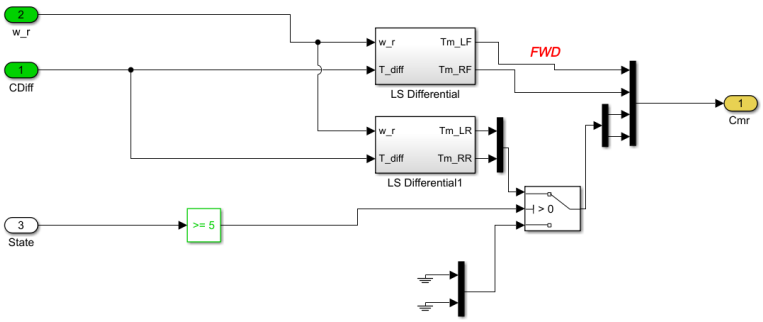

Figure 2.13: Simulink scheme of the Braking System

realistic one: it is just a sort of "simplification" we used in order to reduce the number of command variables while still leaving most of the complexity of the Full Vehicle model.

2.5.4 Vehicle and Terrain

This Section will deal with the two main topics of the Full Model, i.e. the Degrees of Freedom of the Car Body and the Road-Wheel contact with the Pacejka tire model. In discussing this section, we will follow the same approach followed by the Full Vehicle model, i.e. from the ground contact to the equilibrium equations of the car body.

Road-Wheel contact

Therefore, we are going to address the whole Dynamics of the Full vehicle starting from the Kinematics of the wheel: in fact, the forces acting on the tyres - which are the main interface between the Car and the Ground [65][66] - are arising from the kinematics of the tyre, like the side slip angle and the longitudinal slip [59]. As already said in Section 2.5.3, the wheel rotational speed ω_w is the base of the speed of the engine ω_{em} as well (at least the front ones, given that our vehicle is FWD) and so we start from the writing of the dynamic equation regarding each single wheel:

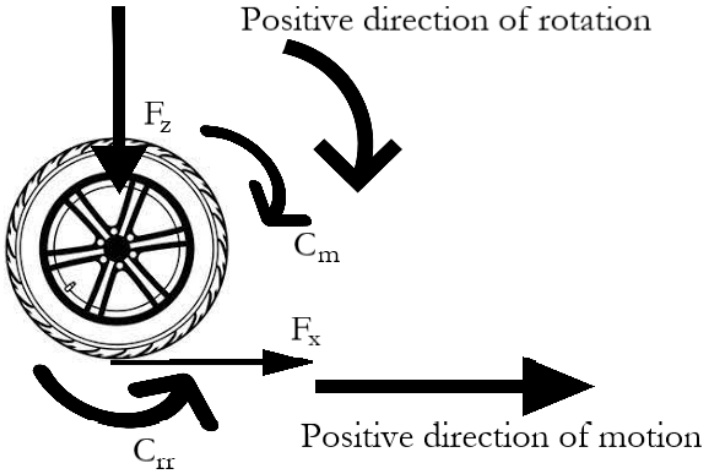

Figure 2.14: Illustration of the equation of motion of the wheel

$$I_w\dot{\omega} = C_m - C_{rr} - F_x R_w \tag{2.18}$$

where C_m is the traction torque coming from the Propulsion, ω_w is the rotation speed of the wheel and C_{rr} is the torque caused by the rolling resistance of the wheel; this is caused by an offset between the vertical load F_z and the ground reaction, due to histeretic phenomena in the rubber material, which can however be modelled as only depending on the rotational speed, following a common approach as for example in [67], as

$$C_{rr} = F_z(f_0 + f_2\omega^2) \tag{2.19}$$

Since the vertical load F_z is dependent from the geometry of the vehicle, C_m depends from the acceleration command and the speed of the motor (Equation 2.14, from Equations 2.18 and 2.19, we conclude that per each of the four wheels we have a free variable, namely the ω_w rotational speed.
The longitudinal force is dependent on the rotational speed and the speed of the vehicle through the slip according to the several Pacejka equations (in our Model, we employed the Pacejka equations from 1991 [59]); the longitudinal slip is defined as an adimensional value ranging from -1 (wheel locked and car moving) to 1 (car stopped and wheel rotating in place) according to Equation 2.20

$$\begin{cases} \sigma = 1 - \dfrac{V}{R\omega} \quad for\ \sigma > 0\ (traction) \\ \\ \sigma = \dfrac{R\omega}{V} - 1 \quad for\ \sigma < 0\ (braking) \end{cases} \tag{2.20}$$

in order to have a Vehicle Model as realistic as possible, we accounted for the radial deformation of the tire caused by the vertical load F_z acting on the wheel, hence R is not constant.

$$\sigma((b_1\mu + b_2)\mu)sin\Big(b_0 atan\Big(\Gamma_x - (b_6\mu^2 + b_7\mu + b_8)\cdot(\Gamma_x - atan\Gamma_x\Big)\Big) \tag{2.21}$$

where for compactness we substituted Γ_x

$$\Gamma_x = \frac{(b_3\mu + b_4)\mu \cdot 2.71828^{(-b_5\mu)}}{b_0\Big((b_1\mu + b_2)\mu + \epsilon\Big)}(100F_z + b_9\mu + b_{10}) \tag{2.22}$$

with F_z being reported in kN instead of N and ϵ is a term used in case of nil μ adherence coefficient to avoid computational problem.
The lateral force F_y generated by the tire is obtained through a similar formula, coming from [59] as well, where the main variable linking the movement to the Force is represented by the side slip angle α.

$$F_z\left(\left((a_1\mu^2 + a_2\mu)sin\Big(a_0 atan\Big(\Gamma_y - (a_6\mu + a_7)(\Gamma_y - atan\Gamma_y)\Big)\Big)\right) + a_1 2\mu + a_1 3\right) \tag{2.23}$$

with Γ_y being

$$\Gamma_y = \frac{a_3 sin\left(atan\left(\dfrac{2\mu}{a_4}\right)\right)}{a_0(a_1\mu^2 + a_2\mu + \epsilon}(\alpha + a_9\mu + a_1 0) \tag{2.24}$$

Once we computed both F_x and F_y for a wheel, we need to account for the combined slip through the elliptical model, a common equation which can be found - among

other sources - in [67] and is represented by the following Equation

$$\left(\frac{F_y}{f_{y,max}}\right)^2 + \left(\frac{F_x}{f_{x,max}}\right)^2 = 1 \tag{2.25}$$

These forces were then delayed according to the "Relaxation Length" model [68] in which the Force - computed according to the current slip - is delayed (in opposition to the approach of Pacejka which delays the slip [69]) as follows:

$$\begin{cases} \dfrac{F_{x,rit}}{F_{x,Pac}} = \dfrac{1}{\tau_L s + 1} \\ \\ \dfrac{F_{y,rit}}{F_{y,Pac}} = \dfrac{1}{\tau_T s + 1} \end{cases} \tag{2.26}$$

where τ_L and τ_T are the time constants respectively of the longitudinal transient and the lateral transient, computed as ratio between the speed in the center of rotations and the relaxation lengths.

$$\begin{cases} \tau_L = \dfrac{L_{rel,L}}{V_{cr}} \\ \\ \tau_T = \dfrac{L_{rel,T}}{V_{cr}} \end{cases} \tag{2.27}$$

A detailed study on the relaxation length can be found in [69].
In order to compute the speed of the wheel mentioned in Equation 2.27, we need to extend the concepts discussed in section 2.2: assumed u and v as the components of the speed of the vehicle as of Equation 2.7, the speed of a wheel in the Reference Frame of the vehicle has, as components

$$\begin{cases} u_w = u - \dot{\psi} y_w \\ \\ v_w = v + \dot{\psi} x_w \end{cases} \tag{2.28}$$

where the y_w and x_w are the coordinate of the wheel in the Reference Frame of the vehicle

- Front Left, $x_w = a,\ y_w = +\dfrac{T_f}{2}$

- Front Right, $x_w = a,\ y_w = +\dfrac{-T_f}{2}$
- Rear Left, $x_w = -b,\ y_w = +\dfrac{T_r}{2}$
- Rear Right, $x_w = -b,\ y_w = +\dfrac{-T_r}{2}$

By composing the vector module of the speed of each wheel we can obtain the speed in the center of rotation V_{cr}, while by computing the arctangent of the ratio, we can obtain the side slip of that wheel with respect to the longitudinal axis of the vehicle as in Equation 2.29

$$\beta_w = atan\left(\frac{v_w}{u_w}\right) = atan\left(\frac{v + \dot{\psi}x_w}{u + \dot{\psi}x_w}\right) \tag{2.29}$$

from which we can obtain the sideslip angle of the wheel with respect to its own plane α_w, which is used in the Pacejka model and is obtained in Equation 2.30 (Figure 2.15).

$$\alpha_w = \beta_w - \delta_w = atan\left(\frac{v + \dot{\psi}x_w}{u + \dot{\psi}x_w}\right) - \delta_w \tag{2.30}$$

Car Body

After discussing the dynamics of the wheels and the 4 DOFs associated with them, we can now move on to the part of the Full Vehicle Model dealing with the car body and its DOFs; a common choice as the Vehicle Model is the 14DOF [70][71]. However, we chose to use a 10 DOFs Model; it is to be noted that such term can indicate two different extensions of the 6 DOF Model:

- A Model in which - to the 6 DOFs have been added 4 DOFs linked to the vertical travel of the wheels, let it be $[z_1...z_4]$ or $[z_f z_r \phi_f \phi_r]$ [72], which is useful to analyze the comfort of the passenger in the NVH field.
- A Model in which - to the 6 DOFs have been added 4 DOFs linked to the rotation of the wheels [73], which sacrifices the vertical motion anbalysis (i.e. the comfort analysis) in order to be able to analyze the dynamic of each wheel.

As we want to deal with a precise study of the dynamic of the vehicle, we employed the second of these. Since we have already discussed the 4 added DOFs in the **Road-Wheel Contact**, we are now going to list here the 6 DOFs of the simple 6 DOF model.

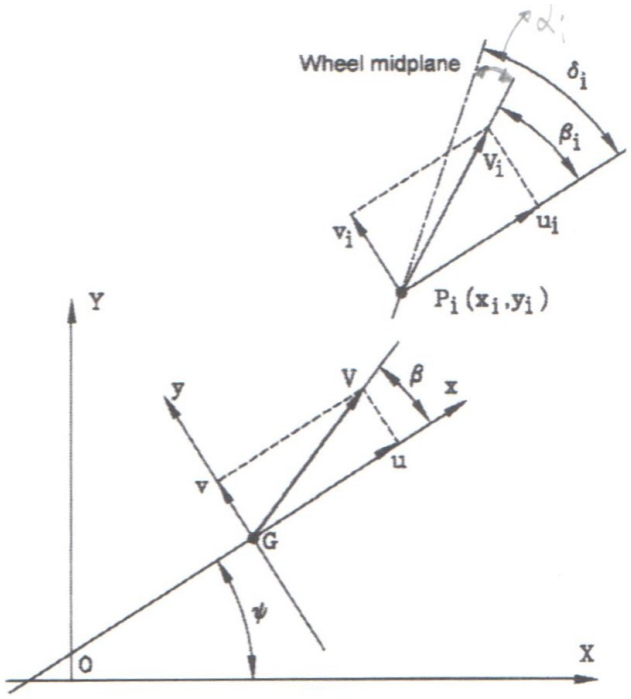

Figure 2.15: Figure 25.16 of [58], displaying the computation of the sideslip angle of the wheel

- Global abscissa X
- Global ordinate Y
- Global vertical position Z
- Roll angle ϕ
- Pitch angle θ
- Yaw angle ψ

Even though we did not account for the vertical displacements of the wheels, yet we could account for the vertical load of every single wheel, important since F_z is an important factor in the computation of both F_x and F_y (Equations 2.21 and 2.23).
The vertical load on every wheel was computed as a sum of **4** different components

$$F_{z,w} = F_{z,st} + \Delta F_{z,acc} + \Delta F_{z,aero} + \Delta F_{z,lat} \tag{2.31}$$

where the static vertical load is given by

$$\begin{cases} F_{z,stFL} = F_{z,stFR} &= \dfrac{mgb}{2L} \\ \\ F_{z,stRL} = F_{z,stRR} &= \dfrac{mga}{2L} \end{cases} \tag{2.32}$$

The load transfer due to the longitudinal acceleration is

$$\begin{cases} \Delta F_{z,accFL} = \Delta F_{z,aeroFR} &= \dfrac{-m\dot{u}H_g}{2L} \\ \\ \Delta F_{z,accRL} = \Delta F_{z,aeroRR} &= \dfrac{m\dot{u}H_g}{2L} \end{cases} \tag{2.33}$$

The load transfer due to the aerodynamic drag, which is applied not in the center of gravity but in the **aerodynamic center**

$$\begin{cases} \Delta F_{z,aeroFL} = \Delta F_{z,aeroFR} &= -F_{x,aero}\dfrac{H_a}{2L} \\ \\ \Delta F_{z,aeroRL} = \Delta F_{z,aeroRR} &= F_{x,aero}\dfrac{H_a}{2L} \end{cases} \tag{2.34}$$

with the aerodynamic drag obtained from the longitudinal component u of the speed

$$F_{x,aero} = \frac{1}{2}C_x A_f \rho_{air} u^2 \tag{2.35}$$

while the load transfer due to the lateral forces are

$$\begin{cases} \Delta F_{z,latFL} = -\dfrac{\phi K_f + \dot{\phi} C_f + (F_{y,FL} + F_{y,FR})(H_g - H_{roll,f}}{T_f} \\ \\ \Delta F_{z,latFR} = \dfrac{\phi K_f + \dot{\phi} C_f + (F_{y,FL} + F_{y,FR})(H_g - H_{roll,f}}{T_f} \\ \\ \Delta F_{z,latRL} = -\dfrac{\phi K_r + \dot{\phi} C_r + (F_{y,RL} + F_{y,RR})(H_g - H_{roll,r}}{T_r} \\ \\ \Delta F_{z,latRR} = \dfrac{\phi K_r + \dot{\phi} C_r + (F_{y,RL} + F_{y,RR})(H_g - H_{roll,r}}{T_r} \end{cases} \tag{2.36}$$

Despite not appearing directly in Equation 2.36, lateral acceleration a_y plays a major role in the load transfer due to lateral forces, through the roll DOF ϕ. Since

we are going to neglect the pitch movement, the last DOF we need to discuss is the roll ϕ; we start from the computation of the lateral acceleration in the Roll Center, which is different from the one in the center of gravity in case the two centers do not correspond (i.e. if H_{roll} is not nil).

$$a_{y,RC} = a_{y,G} + \ddot{\phi} H_{roll} cos\phi \tag{2.37}$$

from which we can compute the roll acceleration $\ddot{\phi}$

$$\ddot{\phi} = \frac{\left(m a_{y,RC} H_{roll} cos\phi\right) + \left(mgH_{roll} sin\phi\right) - \left(K_{roll}\phi + G\dot{\phi}\right)}{J_x + mH_{roll}^2} \tag{2.38}$$

2.6 The vehicle's Sensing Equipment

This Section is going to be devoted to the exposition of the Sensing Equipment we supposed to install on our EgoVehicle; even though we virtually had infinite possibilities, given that this thesis work done entirely on Simulink, allowing us to install as many sensors as we wanted, **without any monetary constraint,** and that such sensors could be as powerful as we wanted: while a real radar has Range Limits (for example a general range limit for Automotive application is 200 m or lower [74][75]).
However, since our aim has always been to develop something that in a near future can be used in real-life application if not even at a mass production level, we took as a general rule to employ only realistic sensors and to try and reduce the number in order to keep the costs low.
Here is the list of all the Sensors we have installed:

- Camera, used to detect the Lane Boundaries.
- Central Radar, used to detect Leading vehicles.
- Side Radar, used to detect Oncoming vehicles.
- BlindSpot Radar, used to detect the completed Overtake.
- *Only in a Beta state: Left BlindSpot Radar*, used to detect an Overtaking vehicle.

In the following sections, we are going to discuss each sensor, one by one with their specifics and their purpose.

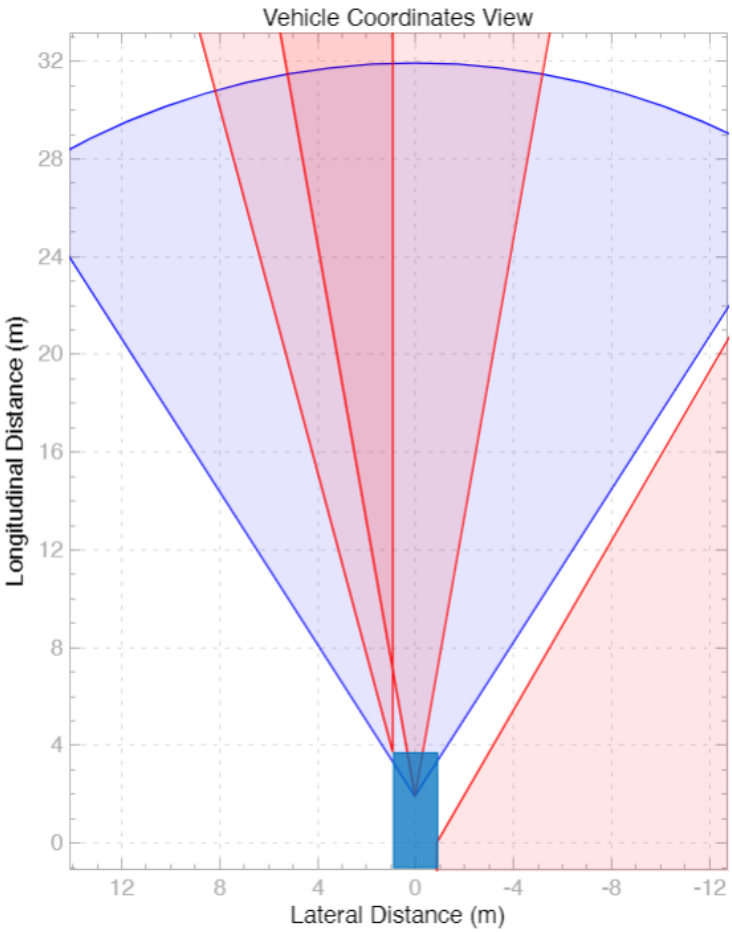

Figure 2.16: Map of the Sensors mounted on the EgoVehicle zoomed to see the relative positioning. Colour code comes from Simulink and is **blue Cameras** and **red for Radars**

2.6.1 Camera

The Camera was used for both Lane Detection and Object Detection, even though its main domain is the former, as the Object Detection (of the leading vehicle) is performed in cohabitation with the Central Radar of which we will discuss later. For what concerns the Lane Detection, instead, the whole burden falls on the Camera [21] [33], as Radar is not able to distinguish colours and therefore it would not be capable of detecting the white lines delimiting the Lane, as well as geometric limitations meaning a Lidar installed in place of the Radar would only be able to recognize lane boundaries at some tens of meters from the Ego Vehicle. We are here listing the main parameters of the Camera, most of which were kept as default from the Simulink **Vision Detection Generator** block.

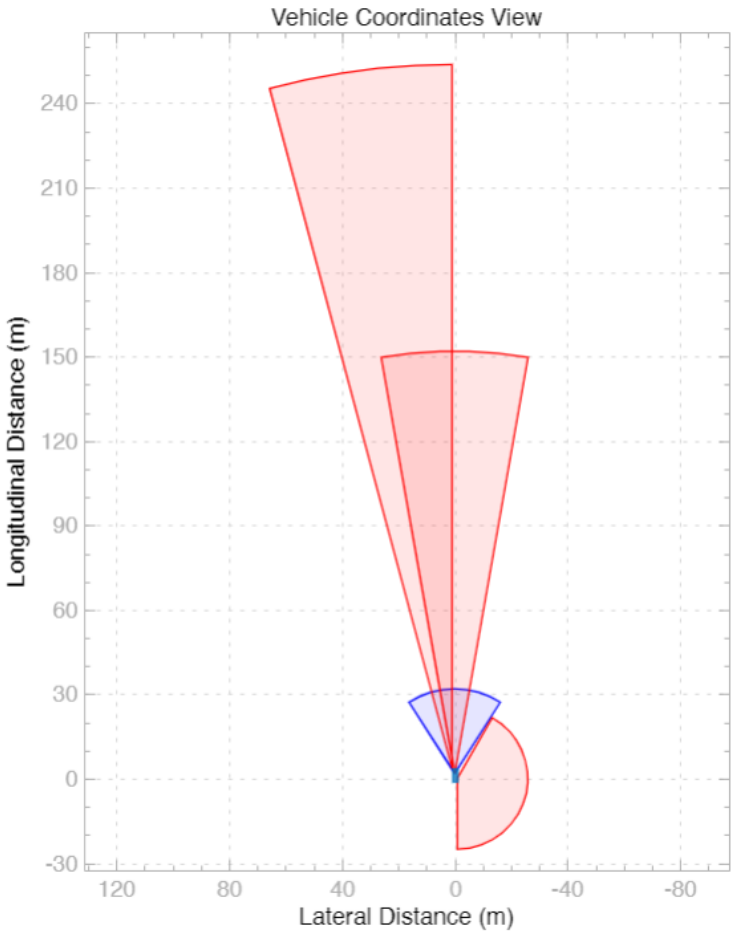

Figure 2.17: Map of the Sensors mounted on the EgoVehicle from a larger distance to see all the Sensors ranges

- **Types of detections:** Lanes and objects.
- **Required interval between sensor updates:** 0.1 s.
- **Required interval between lane detection updates:** 0.1 s.
- **Sensor's (x,y) position:** [1.9 m, 0 m].
- **Sensor's height:** 1.1 s.
- **Yaw angle of sensor mounted:** 0°.
- **Pitch angle of sensor mounted:** 1°.
- **Roll angle of sensor mounted:** 0°.

- **Coordinate system used to report detections:** Ego Cartesian.
- **Maximum detection range:** 30 m.
- **Smoothing filter noise intensity:** 5 m/s^2.
- **Maximum detectable object speed:** 100 m/s.
- **Maximum allowed occlusion for detector:** 0.5.
- **Minimum detectable image size of an object:** [5 px, 5 px].
- **Probability of detecting a target:** 0.9.
- **Number of false positives per image:** 0.1.
- **Minimum lane size in image:** [20 px, 1 px].
- **Accuracy of lane boundary** 1 px.
- **Focal length:** [500 px, 800 px].
- **Optical center of the camera:** [320 px, 240 px].
- **Yaw angle of sensor mounted:** [480 px 640 px].
- **Radial distortion coefficients:** [0 0].
- **Tangential distortion coefficients:** [0 0].
- **Skew of the camera axes:** 0.

2.6.2 Central Radar

The Central Radar is tasked with the detection and tracking of an eventual leading vehicle and output the relative distance ΔX and the relative speed ΔV so that we can perform the Decision Making.
Most of the parameters were left as the default ones from the Simulink **Radar Data Generator** block, we are here reporting the most important ones.

- **Update Rate:** 10 Hz.
- **Sensor mounting relative to Vehicle Origin:** [1.9 m, 0 m, 0.2 m].
- **Sensor rotation relative to Vehicle Frame:** [0°, 0°, 0°].
- **Azimuth resolution:** 1°.

- **Range resolution:** 2.5 m.
- **Range rate resolution:** 0.5 m/s.
- **Angular field of view [Azimuth, Elevation]:** [20°, 5°].
- **Range Limits:** [0 m, 150 m].
- **Range rate limits:** [-100 m/s 100 m/s].
- **Detection probability:** 0.9.
- **False alarm rate:** 1e-06.

2.6.3 Side Radar

The Side Radar is tasked with the detection and tracking of oncoming vehicles and this guided us in its location and in the definition of its parameters.
In order to have a clear sight of the leftmost Lane, which is where we suppose to be detecting oncoming vehicles, we positioned this Radar on the front bumper of the EgoVehicle, on the leftmost part; meanwhile, we also rotated the Radar with respect to the non-inertial Reference Frame of the EgoVehicle, in order to have a detection area fully on the left of the vehicle.

- **Update Rate:** 10 Hz.
- **Sensor mounting relative to Vehicle Origin:** [**3.7 m, 0.9 m,** 0 m].
- **Sensor rotation relative to Vehicle Frame:** [**7.5°,** 0°, 0°].
- **Azimuth resolution:** 1°.
- **Range resolution:** 2.5 m.
- **Range rate resolution:** 0.5 m/s.
- **Angular field of view [Azimuth, Elevation]:** [15°, 5°].
- **Range Limits:** [0 m, 250 m].
- **Range rate limits:** [-100 m/s 100 m/s].
- **Detection probability:** 0.9.
- **False alarm rate:** 1e-06.

2.6.4 Blind Spot Radar

The Radar located near the Rear Right Wheel was dubbed "Blind Spot Radar" because of its purpose, to detect any object in the rear right blindspot, which in Right-Hand Traffic countries (representing 65% of the world population and 70% of the world roadways [76]) is where generally an overtaken vehicle is located. This is the actual purpose of such Radar, as will be discussed in Section 2.7.6. Below are reported the main parameters of the radar.

- **Update Rate:** 10 Hz.
- **Sensor mounting relative to Vehicle Origin:** [0 m, **-0.9 m, 0.2 m**].
- **Sensor rotation relative to Vehicle Frame:** [**-105°,** 0°, 0°].
- **Azimuth resolution:** 1°.
- **Range resolution:** 2.5 m.
- **Range rate resolution:** 0.5 m/s.
- **Angular field of view [Azimuth, Elevation]:** [**150°**, 5°].
- **Range Limits:** [0 m, **25 m**].
- **Range rate limits:** [-100 m/s 100 m/s].
- **Detection probability:** 0.9.
- **False alarm rate:** 1e-06.

The main parameter is the very wide azimuth range, which was needed in order to be able to detect and track an overtaken vehicle. Since the overtaken vehicle will still be close to us, a long range was not needed, therefore we employed a short range of just 25 m, which is still sufficient to detect when we have completed the overtake. The relative yaw angle was chosen in order to have the boundary of the detection zone parallel to the longitudinal axis of the EgoVehicle.

2.6.5 Left BlindSpot Radar

As stated before, this Radar is only in a Beta-state, i.e. we have started some experimentation with it, but for reasons of simplicity and computational cost we have decided to leave it out for now. Its purpose was to detect if some vehicle behind us had already begun the Overtaking maneuver, meaning that we could not begin it ourselves in order to avoid an accident. Even if it was not part of our Sensor Configuration (hence why it is not present in Figures 2.16 and 2.17), we

decided to discuss it as well, since it can be used for further developments. Below are reported the main parameters of the radar.

- **Update Rate:** 10 Hz.
- **Sensor mounting relative to Vehicle Origin:** [0 m, **0.9 m, 0.2 m**].
- **Sensor rotation relative to Vehicle Frame:** [**135°,** 0°, 0°].
- **Azimuth resolution:** 1°.
- **Range resolution:** 2.5 m.
- **Range rate resolution:** 0.5 m/s.
- **Angular field of view [Azimuth, Elevation]:** [**90°**, 5°].
- **Range Limits:** [0 m, 150 m].
- **Range rate limits:** [-100 m/s 100 m/s].
- **Detection probability:** 0.9.
- **False alarm rate:** 1e-06.

2.7 The handling of the Perception data

After having discussed the Vehicle Model we employed during the simulations, the Sensing Equipment we "mounted" on it and the Scenarios on which we performed the testing, we can finally move to the description of the Software pipeline that we developed in order to be able to take decisions and plan our path.

2.7.1 How to deal with dangerous situations

At the beginning of this Section on the handling of the Perception pipeline data, we want to describe the OvtCounter variable and its motivations.
The reason for the creation of this lies in the need for something to take into account the fact that we have previously seen an oncoming vehicle; it is a common situation, in fact, to see from distance an oncoming vehicle which is not seen anymore later even if it is still there: this can happen if we are driving on a long winding right curve and, as the oncoming vehicle moves closer, it is not seen anymore by us, as our sight is occluded by a leading vehicle. A human driver would remember that he has seen the oncoming vehicle before his view became occluded by the leading

vehicle, but how to deal with this for a Decision Making Algorithm?
In order to account for this "memory" feature, we created a variable called ***OvtCounter***, which gets raised whenever we see an oncoming vehicle or we see the Lane marker being dashed as in the following Algorithm.
Since the function is running at 10 Hz like the rest of the Simulink model, it is

Algorithm 2 OvtCounter

function OVTCOUNTER($Oncoming, dashed$)
  if $Oncoming|dashed == 0|UnderOvt$ **then** ▷ Where $UnderOvt$ is in a beta state, like the Left BlindSpot Radar
    $OvtCounter = 30$
  else
    $OvtCounter = max\big(0, counter - 1\big)$
  end if
end function

clear that it acts as a 3 seconds delay. The value of 3 seconds was chosen empirically by experience of the thesis author on his daily commute, where he monitored the average and longest duration of leading vehicle occlusions of oncoming vehicles, in order to come up to a value of time which both ensures safety and does not hinder Overtaking Chances too much.

2.7.2 Flags

In order to perform the decision making, which will be later discussed in Section 2.11 we introduced two Flags, which have a *binary* logic:

- **Leading Flag** which can assume the value 0 to indicate that there is no leading vehicle and 1 to indicate that there is a leading vehicle.
- **oncoming vehicle** which can assume the value 0 to indicate that there is no oncoming vehicle and 1 to indicate that there is a oncoming vehicle.

The **Leading Flag** is coming from the Center Radar data analysis (Section 2.7.5), while the Oncoming Flag can come from both the Center Radar or the Side Radar data analysis, through a **maximum** block. (Figure 2.18)
A third Flag, the **Overtaken** flag, is obtained from the Right BlindSpot Radar and is used in the Finite State Machine (Section 2.11) to move from the State 3 (OVERTAKE) to the State 6 (GO BACK).
One further Flag, which is not simply binary as the previous ones, is the **Dashed**

Flag, which is used to know if the Lane Marker is dashed (= overtaking is possible) or continuous (= overtaking is prohibited). Below is the series of possible values assumed by this Flag:

- Dashed = 0: the Lane Markers are all Continuous
- Dashed = 1: the Left Lane Marker is dashed = we are in our Own Lane and we can overtake
- Dashed = 2: the Right Lane Marker is dashed = we are in the Overtake Lane.

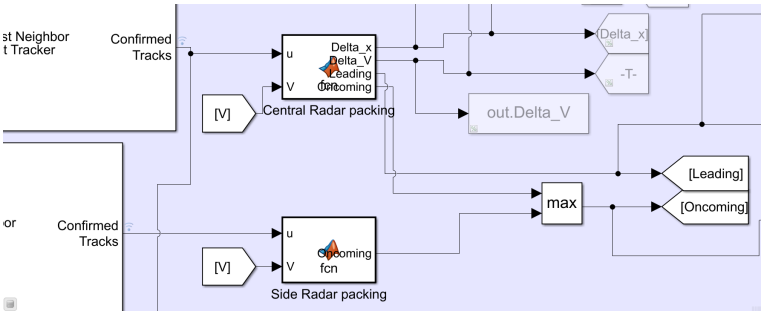

Figure 2.18: "Maximum" block to obtain the Oncoming Flag

2.7.3 GNN Tracker

Before starting to comment Sensor by Sensor, we want to carry out a precisation: before running any function written by us, we applied to the perception pipeline the so-called **Global Nearest Neighbor Multi Object Tracker** [77], which is a preset block in the Matlab Simulink environment; such block performs a tracking of the Lead and oncoming vehicles, collecting the *detections* from the Radars and attributing them through a Nearest Neighbor Classification [78] to a *track*.
The main weakness of this block is that it slows down significantly our Simulink model, as we have observed by running the same simulation and computing the requested time per each of the two simulations through the Matlab tic ... toc command; we know that tic ... toc is not a real time simulation and we can not take these results as a perfect measure, nevertheless, we used it in order to get an order of magnitude of the time taken as well as to compare the computational cost of the system with and without GNN. The simulation was a 175 seconds long simulation on the **ACC_Scenario**. Here are the results:

- Without the GNN blocks, the simulation took 233.5 s, which equates to **133.4%** of the simulated time.

- With the GNN blocks, the simulation took 271.7 s, which equates to **155.3%** of the simulated time.

The GNN increases the required time of about 16%, but since the time without the GNN was still too much to be applied, since even a 90-95% time would be unfit, given that a car's CPU runs much slower than the PC on which we performed the simulations.
Luckily, the main component in terms of computational costs are the Full Vehicle Model itself, the Scenario Reader and the Radar blocks, which in the real vehicle will not be simulated, as they will be real. Because of this, and because of the results that we are going to present now, we decided to employ the GNN regardless of the increase in the computational time.

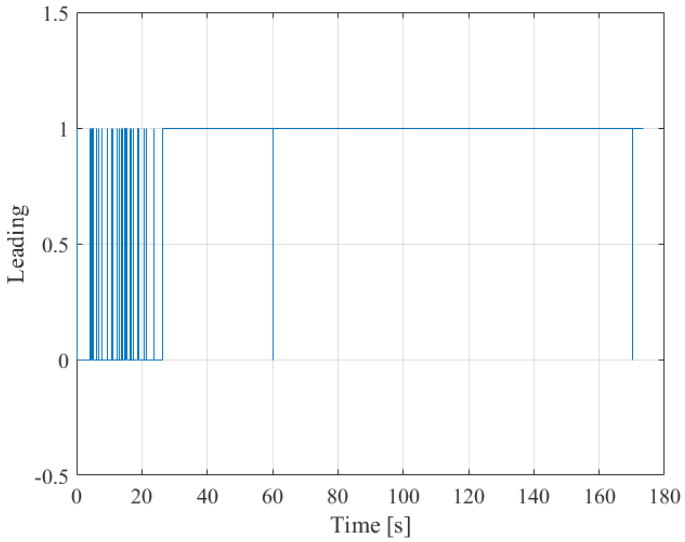

Figure 2.19: Trend of the Leading Flag during the simulation run **without** the GNN tracker

We can immediately notice, through the comparison of the Flags, that the version without the GNN is immensely more noisy and it also sees oncoming vehicles when there are none, therefore making the maneuvers much more dangerous.
The noisy results of the two Flags are of course heavily linked to the noisy results on the radar in general, as can be seen by a rapid comparison of Figures 2.23 and 2.24. It is also to be noted that such simulations were run in an early build of the Model: this can be seen by the fact that even the GNN model is assigning values of ΔX and ΔV to oncoming vehicles, while this is not the case in the final build, which assigns such values only to leading vehicles.
Most notably, the noisy trend in the radar signals is not important just in itself:

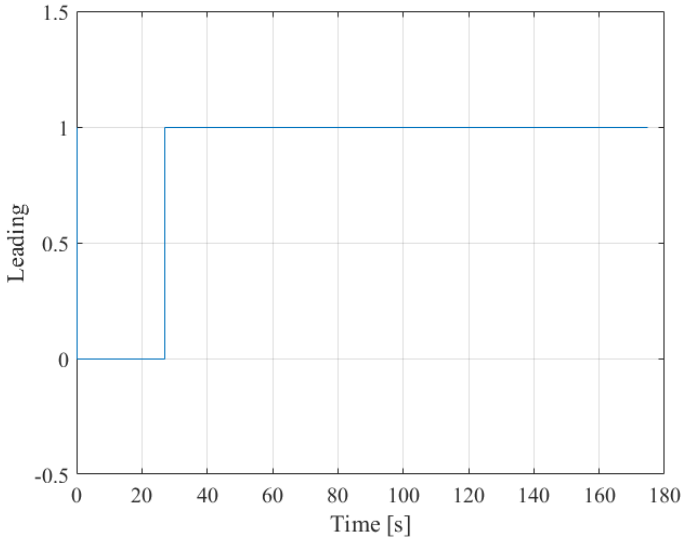

Figure 2.20: Trend of the Leading Flag during the simulation run **with** the GNN tracker

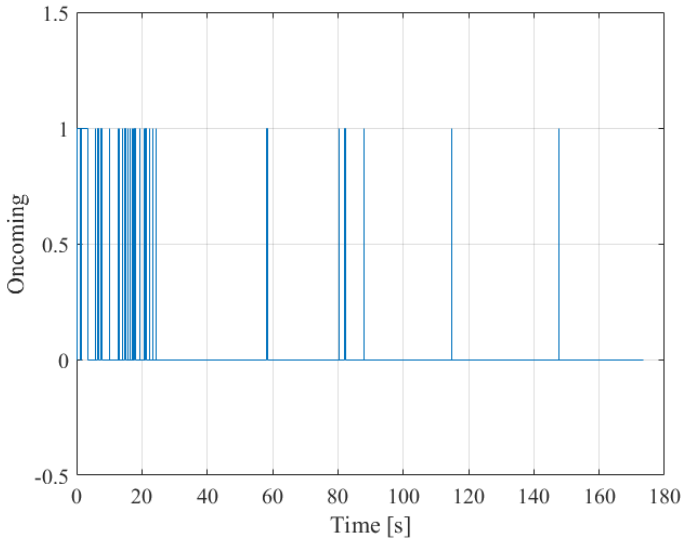

Figure 2.21: Trend of the Oncoming Flag during the simulation run **without** the GNN tracker

the noise in the detection and recognition of Leading/oncoming vehicles and their relative positions and speeds leads to a noisy behaviour in the EgoVehicle itself, as can be seen by the comparison in the trends of Time gap (Figure 2.25 and

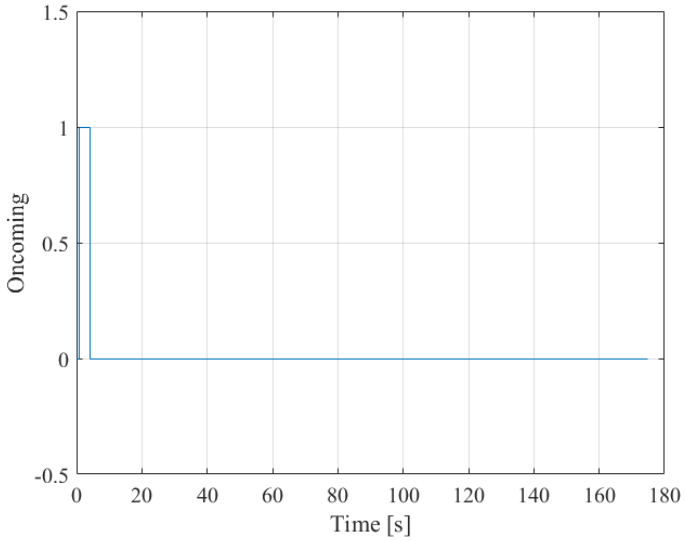

Figure 2.22: Trend of the Oncoming Flag during the simulation run **with** the GNN tracker

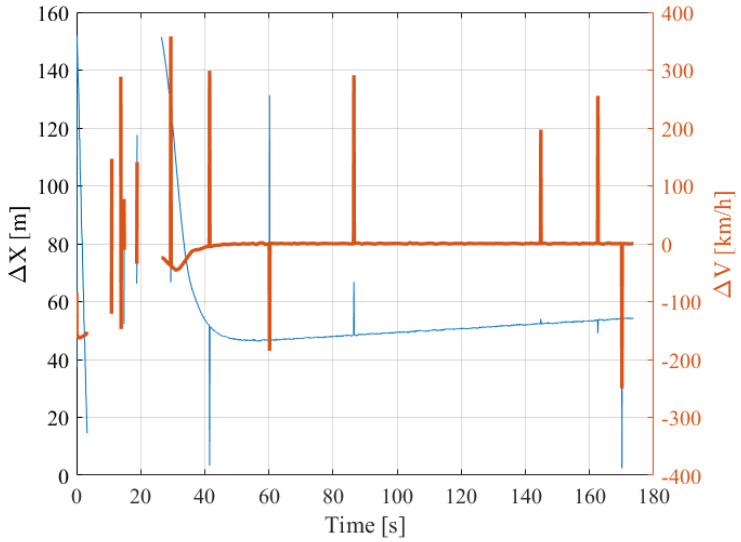

Figure 2.23: Radar signals during the simulation run **without** the GNN tracker

2.26) and - even more importantly for the comfort of the passengers - speed of the EgoVehicle (Figure 2.27 and 2.28).

Below are the thresholds we used for the confirmation and deletion (History logic) of every GNN Tracker:

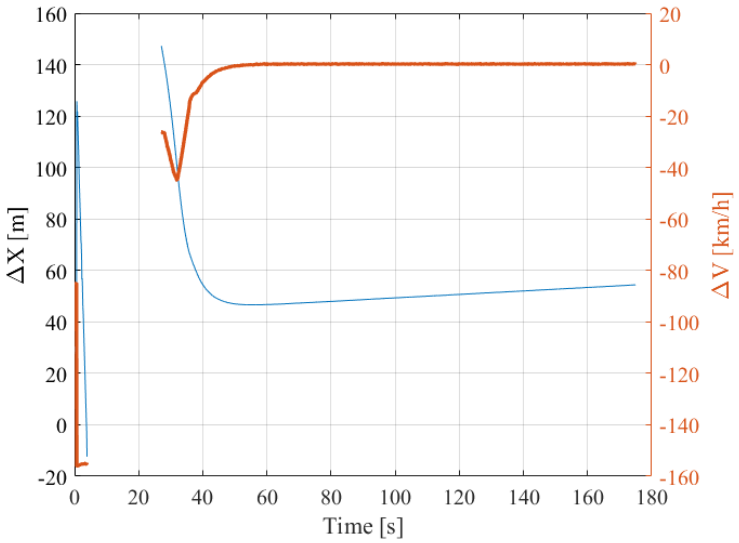

Figure 2.24: Radar signals during the simulation run **with** the GNN tracker

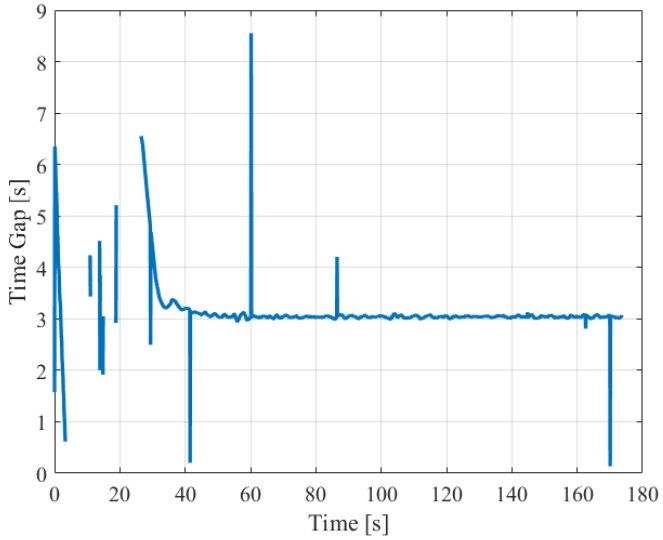

Figure 2.25: Time gap between the EgoVehicle and the lead vehicle during the simulation run **without** the GNN tracker

- GNN on the Detection Concatenation between Center Radar and the Camera:
  - Confirmation threshold [M N]: [5, 7]
  - Deletion threshold [P Q]: [8, 10]

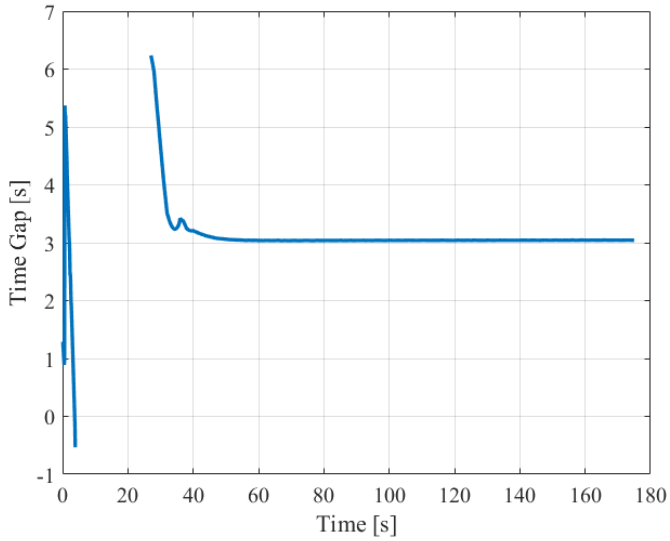

Figure 2.26: Time gap between the EgoVehicle and the lead vehicle during the simulation run **with** the GNN tracker

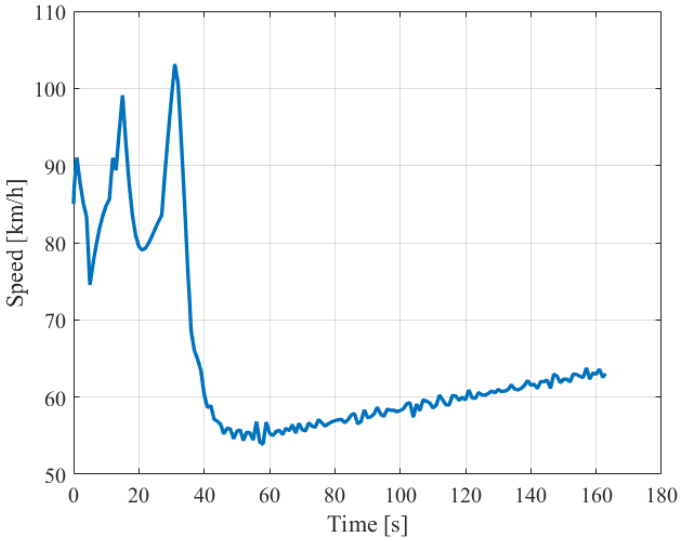

Figure 2.27: Trend of the EgoVehicle speed during the simulation run **without** the GNN tracker

- GNN on the Side Radar
  - Confirmation threshold [M N]: [3, 5]
  - Deletion threshold [P Q]: [6, 10]

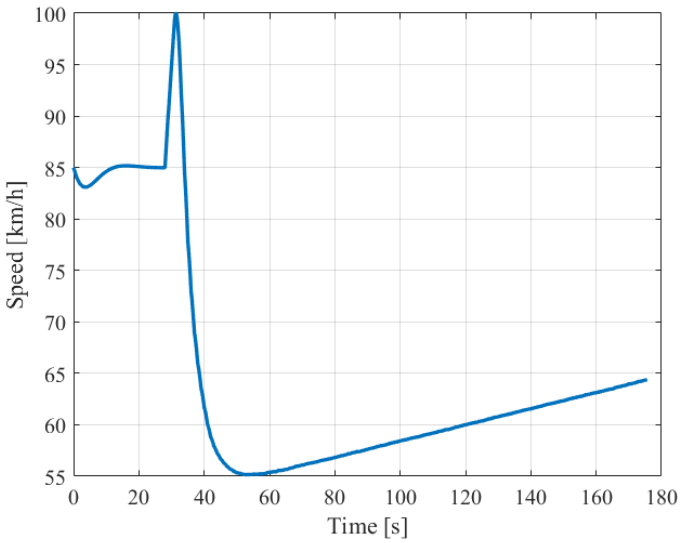

Figure 2.28: Trend of the EgoVehicle speed during the simulation run **with** the GNN tracker

- GNN on the Right BlindSpot Radar
  - Confirmation threshold [M N]: [7, 7]
  - Deletion threshold [P Q]: [6, 10]

2.7.4 The Camera data analysis: Lane Recognition

The first thing we want to point out is that these algorithms we are going to present next have been developed in the Simulink environment and therefore they were designed to extract information from the Matlab structures output by said GNN Tracker; of course, in a real world application with real sensors we would need different logic in unpacking the signals, but what we want to point out is the data analysis logic, which is going to stay untouched by moving to another packing logic. The first thing we do is a for cycle to save up in two vectors the Lateral Offset (*LatOff* vector) and the Heading Angle (*HeadAngle* vector) of every LaneBoundary object, up to the $NumLaneBoundaries^{th}$; in the Matlab Simulink environment, the Lanes are already ordered from left to right, while in a real application we will need to be sorting them: to do this, when we save them up in the vector, we will sort them by the value of the LatError, which is the highest on the left and the lowest on the right. Once this is done, we want to clear the two vectors of every possible NaN elements, in order to have only meaningful elements in those vectors. Because n boundary lanes describe $n-1$ proper Lanes, once we have sorted and

cleared the two vectors we can simply compute the number of Lanes for the vehicles by

$$N_{lanes} = length(LatOff) - 1 \tag{2.39}$$

In case the number of Lanes - for any reason - is less than 1, we output

- **Lane:** 1.
- **OvtOff:** 0.
- **OvtHead:** 0.
- **OwnOff:** 0.
- **OwnHead:** 0.
- **Dashed:** 1.

This is to ensure that the EgoVehicle does not start any Overtaking maneuver and does not try to steer in case the Camera fails for one frame.
If, on the other hand, a valid acquisition has been performed and at least two Boundary Lanes are detected the algorithm goes on: we compute the LatOff and the HeadAngle for each of the lanes, then we find which is the closest Lane centerline, therefore attributing the EgoVehicle to that Lane.
Once we have detected the Lane in which we are, we need to check the value of the BoundaryType variable corresponding to its left boundary: if it is 2, it means that the boundary is **dashed**, therefore an overtaking is possible. Below we report the PseudoAlgorithm. It is to be noted that in case we see just one lane or even

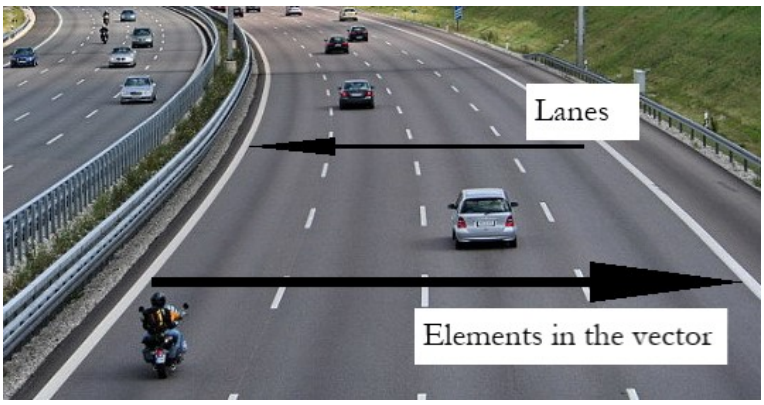

Figure 2.29: Coordinate system of the Lanes as used in this data analysis algorithm

worse we see a single Lane Boundary, thus outputting a number of Lanes of 0,

Algorithm 3 Camera data analysis

function DATA ANALYSIS(u) ▷ Where u is GNN_{output}
  for i from 1 to u.NumLaneBoundaries **do**
    append u.LaneBoundaries(i).LateralOffset to ***LatOff*** vector
    append u.LaneBoundaries(i).HeadingAngle to ***HeadAngle*** vector
  end for
  clear ***LatOff*** and ***HeadAngle*** from NaN
  NumLanes = length(***LatOff***)-1
  if $NumLanes < 1$ **then**
    Lane = 1
    OvtOff = 0
    OvtHead = 0
    OwnOff = 1
    OwnHead = 0
    Dashed = 1
  else
    CenterOffset = $\Big(LatOff(1:end-1) + LatOff(2:end)\Big)/2$
    LanesWidth = $LatOff(1:end-1) - LatOff(2:end)$
    CenterHead = $\Big(HeadAngle(1:end-1) + HeadAngle(2:end)\Big)/2$
    if NumLanes = 1 **then**
      Lane = 1;
      OvtOff = NaN;
      OvtHead = NaN;
      OwnOff = CenterOffset;
      OwnHead = CenterHead;
      Dashed = 1;
    else
      index = $min\Big(abs(CenterOffset)\Big)$ ▷ we find the Lane whose center is the closest to the EgoVehicle
      Lane = NumLanes+1-index ▷ to further understand the indices, check Figure 2.29
      OvtW = LanesWidth(max(NumLanes-Lane,1),1);
      OvtOff = CenterOffset(max(NumLanes-Lane,1),1);
      OvtHead = CenterHead(max(NumLanes-Lane,1),1);
      OwnW = LanesWidth(max(NumLanes-Lane+1,1),1);
      OwnOff = CenterOffset(NumLanes-Lane+1,1);
      OwnHead = CenterHead(NumLanes-Lane+1,1);
      if u.LaneBoundaries(NumLanes+1-Lane).BoundaryType == 2 **then**
        Dashed = 1;
      else if u.LaneBoundaries(NumLanes+2-Lane).BoundaryType == 2 **then**
        Dashed = 2;
      else
        Dashed = 0;
      end if
    end if
  end if
end function

we imposed the output of the **dashed** flag to 1: this has been needed in order to avoid sharp aborts of Overtaking maneuvers, which would be caused by the car not seeing enough Lane Boundaries because the Camera is occluded by the presence of the Lead vehicle.

2.7.5 The Front Radars data analysis: Leading and oncoming vehicles

In this Section we are going to discuss the data analysis for the two Front Radars, the central one - mainly but not only in task of tracking the leading vehicle - and the Left side one, which is tasked with the tracking of oncoming vehicles.

Central Radar

This radar is mainly tasked with the recognition of the leading vehicle and its tracking; the main way to discriminate a tracked Vehicle between a Leading and oncoming vehicles lies in the value of the ΔV relative speed: if the value is lower than the opposite of the speed of the EgoVehicle, it means that the car is Oncoming, while the opposite is true in case the tracked Vehicle is Leading.
However, it is to be noted that, in order to avoid misrecognitions of lead vehicles and Parked Vehicles as oncoming vehicles, we imposed that, to be recognized as an oncoming vehicle, the detection must have a relative speed below $-1.05V$ Because the GNN can output more than one track with just a single Vehicle, the selection of the best detection is tied to the one being closest to the EgoVehicle longitudinal axis: once the closest Track is found, we save its Δx and ΔV and at the end we also raise eventual Leading and Oncoming flags. Once the ΔX and ΔV have been obtained and the flags have been raised or dropped, we still need to correct the value of ΔX; why do we need to do this? The reason lies in the fact that - with the Simulink block - we obtain the relative distance and velocity with respect to the Center of Gravity of the EgoVehicle: this means that if the ΔX is around 3 m, it actually means that we are colliding with the rear bumper of the lead vehicle. Because of how real Radars work, this further passage will not be needed in the real application.
Another thing we want to point out is that it looks like the Central Radar is not capable of detecting an oncoming vehicle if there is also a leading vehicle: in fact, in this case, the vehicle with the lowest (in absolute value) θ angle would of course be the Leading one. This is indeed true, but is not a problem, since the oncoming vehicle would be detected by the Side Radar; the ability to output a raised Oncoming Flag from this radar has been included only for the case in which we are in the Oncoming Lane and there is an oncoming vehicle, since the Side Radar would be looking outside of the road and therefore would not recognize any

Algorithm 4 Central Radar data analysis

function PostProcess(u, Lane) ▷ Where u is GNN_{output}
Initialize θ_{min} to 10e6
Initialize both flags to 0
for i from 1 to u.NumTracks **do**
Retrieve $x, y, \Delta X \Delta V$ from u.Tracks(i)
$\theta = min\left(atan2\left(\frac{y}{x}\right)\right)$
if $\theta < \theta_{min}$ **then**
$\theta_{min} = \theta$
Save up ΔX and ΔV from u.Tracks(i)
end if
end for
if Lane==1 **then**
if $\Delta V > -V$ **then**
Leading = 1
Output Leading, Oncoming, ΔX and ΔV
else
Oncoming = 1
Output Leading, Oncoming. We decided not to output ΔX and ΔV for this branch, leaving only the initialized NaNs
end if
else ▷ This means we are in the Overtake Lane, probably
if $\Delta V < -\mathbf{1.05}V$ **then**
Oncoming = 1
Output Leading, Oncoming. We decided not to output ΔX and ΔV for this branch, leaving only the initialized NaNs
end if
end if
end function

Vehicle.

Side Radar

The Side Radar is tasked with the recognition of oncoming vehicles, as already stated in Section 2.6.3, and this task has guided us in the location of said sensor. In a way similar to what was done for the Central Radar, the discriminant between a Leading and oncoming vehicle is represented by the ΔV relative speed, which is of course lower than the opposite of the EgoVehicle speed only in case the tracked Vehicle is Oncoming.
By the way, it is important to note that this Radar is not capable of recognizing a leading vehicle, i.e. it does not output a Leading Flag value, since this is only controlled by the Center Radar.

Algorithm 5 Side Radar data analysis

function PostProcess(u) ▷ Where u is GNN_{output}
  Initialize θ_{min} to 10e6
  Initialize Oncoming flag to 0
  for i from 1 to u.NumTracks **do**
    Retrieve $x, y, \Delta X \Delta V$ from u.Tracks(i)
    $\theta = min\left(atan2\left(\frac{y}{x}\right)\right)$
    if $\theta < \theta_{min}$ **then**
      $\theta_{min} = \theta$
      Save up ΔX and ΔV from u.Tracks(i)
    end if
  end for
  if $\Delta V < -V$ **then**
    Oncoming = 1
    Output Leading, Oncoming, ΔX and ΔV
  else
    Oncoming = 0
    Output Leading, Oncoming. We decided not to output ΔX and ΔV for this branch, leaving only the initialized NaNs
  end if
end function

2.7.6 The Rear Radars data analysis: Overtake completed and being overtaken

This Section deals with the data analysis function that is applied to the Right BlindSpot Radar, in charge of the Overtake Flag as well as the one which would be applied to the *Left BlindSpot Radar*, if it was in action and already "tested" in our Scenarios.
On the contrary, the *Left BlindSpot Radar* would be raising a **UnderOvt Flag**, to be used for the **OvtCounter** of Section 2.7.1, as it is unsafe to perform a Lane Change to be overtaking, if we are being overtaken ourselves. The way the data analysis on the Right BlindSpot Radar works is quite simple, we find the track which is located the most forward and check its relative position to our vehicle, if this is far behind enough, we can assume that the Overtaken flag can be raised. The fact that this data analysis takes into account the Central Radar too is linked to what we need in order to successfully perform a double Overtake in series, as will be explained in Section 3.9, where we deal with this kind of maneuvers.

2.8 The Control of the Vehicle

The vehicle receives a sets of two commands, the Acceleration command in percentage $[-100\ 100]$ and a steering angle on the ground in radians $\left[\frac{-35*\pi}{180^\circ}\frac{-35*\pi}{180^\circ}\right]$, as already was discussed in Section 2.5. We also want to remember what we also stated about braking, notice Figure 2.13: for sake of simplicity, we omitted a braking command, in order to have only a command and in cases where a heavy braking is needed (States 5 6 and 7), we assume to send the same torque to the Front Differential as well as the Rear Differential.

2.8.1 The Cruise Control

Among the Driver Assistance Systems, the Cruise Control is probably both the oldest and the most widely applied systems; in fact, while some speed regulators were already used in the late XVIII century by James Watt with his flyball governor [79][79], a Cruise Control System as we know it was developed in the late 1940s [80][81], even though simpler applications were already around at the beginning of the XX century.
The working of the base Cruise Controller is quite simple: the difference between the actual EgoVehicle speed is computed and then fed to a PID controller [82], which outputs the acc command.
In our particular system, the reference speed is the minimum between the speed set by the user and 95% of the speed limit, as of Figure 2.30

Algorithm 6 Right BlindSpot Radar data analysis

```
function PostProcess(u, u2)          ▷ Where u is GNN_output for the Right
BlindSpot Radar and u2 is GNN_output for the Center Radar
    if u.NumTracks > 0 then
        Initialize minX to +1000, speed to NaN
        for i=1:u.NumTracks do
            if u.Tracks(i).State(1) < minx then
                maxX = u.Tracks(i).State(1)
                speed = u.Tracks(i).State(2)
            end if
            if u then.Tracks(i).State(3) > maxy
                maxy = u.Tracks(i).State(3);
            end if
        end for
        for i=1:u2.NumTracks do
            if u2.Tracks(i).State(1) < minx then
                maxX = u2.Tracks(i).State(1)
                speed = u2.Tracks(i).State(2)
            end if
            if u then2.Tracks(i).State(3) > maxy
                maxy = u2.Tracks(i).State(3);
            end if
        end for
        if maxX < -3 && speed < 0 then
            overtaken = 1
        else
            overtaken = 0
        end if
    else
        Overtaken, maxX, minX, maxy = NaN
    end if
end function
```

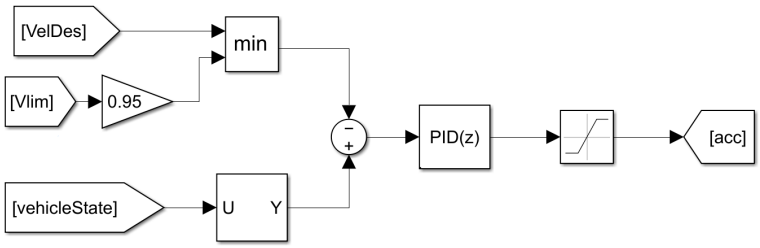

Figure 2.30: The Simulink blocks of our simple Cruise Control system

2.8.2 The Adaptive Cruise Control

Moving forward from the "basic" Cruise controller, its next step and evolution is the so-called Adaptive Cruise Controller, which debuted in the 1990s, with the first application on a road-going vehicle in 1995 on the Mitsubishi Diamante [83].
The working of the ACC is completely different with respect to the traditional CC: rather than aiming at keeping the EgoVehicle speed constant, it tries to keep a constant distance between the EgoVehicle and the lead vehicle; moreover, the variant of ACC that we decided to employ is the so-called Constant Time Gap (CTG) [84][85] which aims at keeping a constant distance between the ego Vehicle and the lead vehicle, but in the meaning of a "time distance" rather than "space distance". Therefore, the desired distance is

$$L_{des} = V_{ego,veh}\Delta t \tag{2.40}$$

with Δt being the desired time gap. The usual value of Time Gap in literature is around 1 s ([84][85]), but we chose to use a 3 seconds gap, because of our need for a high safety, given the EgoVehicle will be unmanned and we want to be sure that no rear-ending occurs: it is true that in our case we would not be subject to the human reaction time (which is the reason for that 1 second value), but a human possesses analysis skills that we could not implement in our System, because of computational cost, which is regulated by the State Machine of Section 2.11. From the comparison between the desired distance L_{des} and the effective distance ΔX we obtain the spacing error δ

$$\delta = L_{des} - \Delta X \tag{2.41}$$

This error is then inserted in a PD (Proportional Derivative) law as found in [84]

$$a = -\frac{1}{\Delta t}(\lambda\delta - \dot{L_{des}}) \tag{2.42}$$

with λ being the main parameter to tune. In Figure 2.31 is shown the block diagram representing such Control in our system.

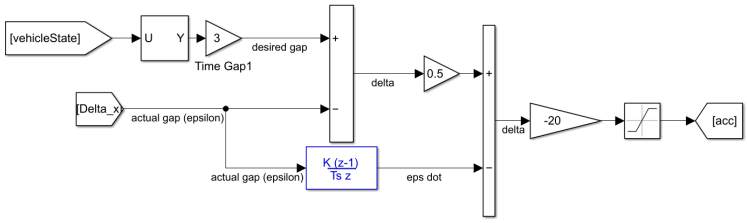

Figure 2.31: The Simulink blocks of our Adaptive Cruise Control system

The main thing which we can see is that - where there should actually be -1/3 as a gain, is instead present a -20; the reason for this is simple, but can be easily missed: while in Equation 2.42 the result a is the desired acceleration of the EgoVehicle (ranging $[-10m/s^2\ 10m/s^2]$), in our case it is the acceleration command, which ranges $[-100\ 100]$ and must therefore be much larger.

2.8.3 The Stanley Controller for Lane Keeping Assist

Before starting the discussion of the Stanley Controller, we want to present the Errors and the Coordinate Systems linked with it; in order to define such Controller for a Lane Keeping Assist, we need to obtain the Cross-Track Error and the Heading Error.

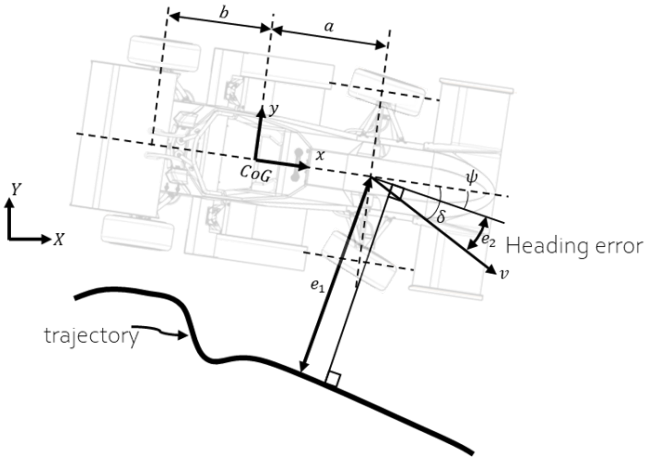

Figure 2.32: The Cross-Track and Hading Error

The Cross-Track Error

With Cross-Track Error is indicated the lateral distance between the center of the front axle and the Trajectory to be followed [35][86], which is - in our case - the

centerline of the Lane we want to stay in (this won't be true for the Overtake maneuver as will be stated later in the corresponding Sections).
The definition of the Cross-Track Error as the difference between the actual position and the reference position

$$e_{ct} = y_{act} - y_{ref} \tag{2.43}$$

meaning that, the Cross-Track Error e_{ct} is positive in consequence of an EgoVehicle which is moved to the left of the Lane compared to its centerline and of course a negative e_{ct} is linked to an EgoVehicle moved to the right. It is to be noted,

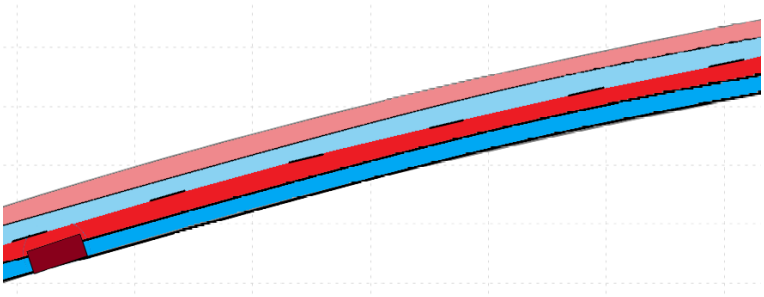

Figure 2.33: An example of the e_{ct} from the Bird'sEye view: in blue the parts of the road where e_{ct} is negative, in red where it is positive. Greyer colours refer to the Overtake Lane

however, that the way we obtain the Cross-Track Error as well as the Heading Error from the Camera data analysis of Section 2.7.4, the Errors we obtain are exactly the opposite of the definition ones.
Let's take as example a case where the EgoVehicle is moved 30 cm towards the left of a Lane 3 m wide, the Cross-Track Error is supposed to be +30 cm, but our Algorithm would receive the following information:

- **Left LaneMarking Lateral Offset:** +1.20 m
- **Right LaneMarking Lateral Offset:** -1.80 m

which would lead to a result of -0.30 m.
This *excursus* was needed in order to show why the block diagram differs from the theoretical formulation.

The Heading Error

In a similar way, the Heading Error is obtained as the difference of the actual yaw position and the target yaw position

$$e_h = \psi_{act} - \psi_{ref} \tag{2.44}$$

Because of the definition, a positive e_h is correlated to a Scenario in which the EgoVehicle is going too much towards the left, with a negative e_h correlated to an EgoVehicle going towards the right; as for the e_{ct}, the errors output from the Camera block are actually their coordinates in the Reference Frame centered on the EgoVehicle origin, **leading to the Errors we get being of opposite sign.**

It is to be noted that this Stanley is applied on an already defined Path, which is implicitly defined if our goal is just to follow the centerline of our Lane and explicitly defined (Sections 2.9 and 2.10) in case we need to change the Lane we are in.
The equation at the base of our System is the "traditional" Stanley Controller, which can be seen for example in [35] and we reported in Equation 2.45. We want to highlight that the values of e_{ct} and e_h used in such Equation are the ones obtained in the Simulink model from the Camera block, i.e. opposite to the real values of e_{ct} and e_h which will be used for reports in Section 3

$$\delta = e_h + atan\left(\frac{Ke_{ct}}{V + V_\epsilon}\right) \tag{2.45}$$

where K is a tuning factor which we imposed as **1.5** while V_ϵ is a small velocity in order to avoid computational errors in case of very low EgoVehicle speed.

2.9 The Overtake Path Planning

In this Section, we are going to discuss the argument giving the title to this whole thesis work: the Path Planning for the Autonomous Overtaking.
As we have already said in Section 1.3.3, the Path Planning algorithm of our choice is going to be the Sigmoid, as was presented by [43][44][45]. The reasons for this choice were, as already stated, the freedom of movement and the low computational cost involved with them, other than the customisability allowed by the factor K.

2.9.1 First step: simplified Sigmoid Planning for DSTP model

To begin the discussion, we are going to report here the equation 1.12 with a slightly different formulation:

$$y = \frac{Y_w}{(1 + e^{K(x - S_m)})} \tag{2.46}$$

This is the equation that we used in the Sensitivity Analysis of Section 2.1, where our EgoVehicle was a simple DSTP model (Section 2.2).

During the first studies on the Sensitivity Analysis we used a simple approach and assumed the coordinates to be **global**: this means that we extracted the position of both the EgoVehicle and the leading vehicle from their own DSTP models and then computed their relative distance and speed starting from these; in order to "mimick" a real world application, we built the Simulink block for the Sigmoid computation with Δx, ΔV and so on as inputs, nevertheless, it was a very primitive attempt.

Even though this attempt was very simple and was working only in Global

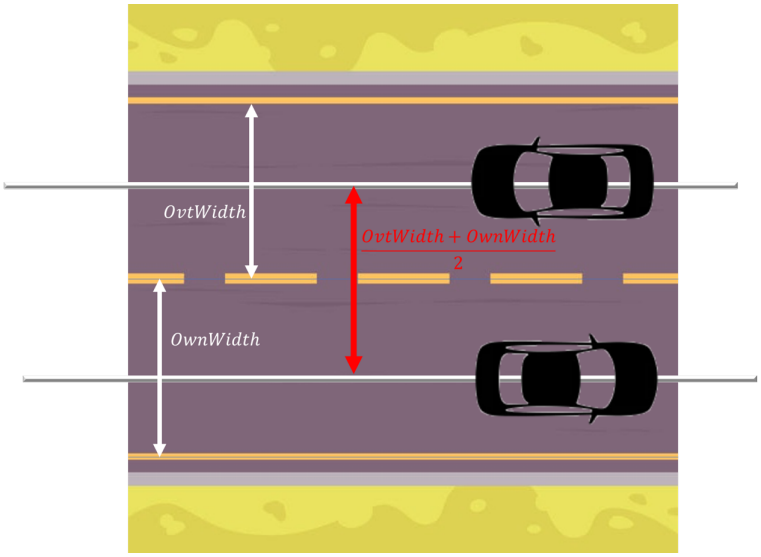

Figure 2.34: A scheme of the different lanes

coordinates and on a straight, it already had *in nuce* some elements that were later kept in the final version:

- First of all, the concept that the Sigmoid Curve is not floating in the middle of nowhere, but is an interpolant curve between the center of the Own Lane and the center of the Oncoming Lane.
- The V_{goal} setting is similar to what we have chosen to do in the State Machine State 3 (OVERTAKE) discussed in Section 2.11.3: in order to minimize the time spent in the Oncoming Lane, we should accelerate up to the Speed Limit while we approach the leading vehicle that we decided to overtake.
- The ***flag* starting at 1** and later dropped to 0 is enabling the block containing this very same function: this means that until the EgoVehicle is close enough to the car in front we keep recomputing new Sigmoid Curves and once we are close enough that we should be already some cm moved to the left ($if \frac{LaneWidth}{1+exp(K\cdot-\xi)} > LaneWidth/100$) we "freeze" the Sigmoid Curve and employ that one. This concept - albeit modified - was carried over to the final model.

Algorithm 7 Simplified Sigmoid

function SIGMOID($V, x, LaneBoundaries, K_{vec}, V_{vec}, \Delta X$) ▷ where K_{vec}, V_{vec} are the interpolants from Section 2.1
$K = interp(V_{vec}, K_{vec}, V)$
$d_{safe} = (V - \Delta V) \cdot 1$ ▷ where ΔV is positive if we are faster
Let x be an array of 333 values from $x - \Delta V \cdot 5$ to $x + \Delta V \cdot 25$
$OwnCenter = LaneBoundaries\{1\}$ ▷ where LaneBoundaries is a struct containing two $333x2$ matrices for the centers of the OwnLane and the OvtLane
$OvtCenter = LaneBoundaries\{2\}$
$LaneWidth = mean(OvtCenter(:,2) - OwnCenter(:,2))$
$\xi = -\Delta X + d_{safe}$
if $\frac{LaneWidth}{1+exp(K \cdot -\xi)} > LaneWidth/100$ **then**
$x = OwnCenter(:,1)$
$path = \frac{LaneWidth}{1+exp(K \cdot -\xi)}$
$Y = OwnCenter(:,2) \cdot (1 - path) + OvtCenter(:,2) \cdot path$
$\theta = atan2(diff(Y), diff(x)$
$Poses = [x, Y, \theta]$
flag = 0
$V_{goal} = V$
else
$Poses = [x, Y, zeros(333,1)]$
flag = 1
Let V_{goal} be the limit speed
end if
end function

As said, however, this approach was completely unfeasible in a model like the one we intended to use for the Full Vehicle testing, where the errors are not directly computed from comparison between the position and the centerline, but obtained through the **Vision Detection Generator** Simulink block, i.e. through a Camera detecting the Lane Boundaries. Because of this, we had to move from this **global** approach to a **local** approach based on the correction of the Controls discussed in Section 2.8.3. Nevertheless, we decided to comment the first solution because of its close relation with what we applied in the end.
As will be later pointed out in Section 2.11.3, as soon as the EgoVehicle exits from State 1 (STAY) to move either in State 2 (WAIT) or in State 3 (OVERTAKE), the car is forced to move towards the left of a quantity equal to $OwnWidth/4$: this will be relevant later on, when discussing our implementation of the Sigmoid Curve.

2.9.2 Definitive model: the Sigmoid for the Full Vehicle model

The main change to enact from the first simplified attempt was to move from **global** to a **local** system of coordinates, given that in a real application it would not make sense otherwise, as there would be no univocal origin and direction and - at the same time - the position of the leading vehicle would be given directly in a local Reference Frame. To do so, we defined a new Reference Frame, in a way similar to the one employed by [44] in its Figure 3 (Figure 2.35 of this thesis work): our Reference Frame is centered in the center of the rear bumper of the lead vehicle with the x axis orthogonal to said bumper and the y axis directed towards the Oncoming Lane, therefore the ΔX relative distance is going to be the x of this Reference Frame, which is displayed in Figure 2.36.
Following the approach of [44], shown in Figure 2.37, one of the constraints that we set for the computation of the Sigmoid Curve was that **when the EgoVehicle is at a distance** $\Delta X = safedistance$ **from the lead vehicle**, the lateral position is on the boundary between the Own Lane and the Oncoming Lane, which would correspond to a $e_{ct} = OwnWidth/2$.
Therefore, we turned the Equation 2.46 into a function $y = f(\Delta x)$ as in Equation 2.47:

$$y = \frac{\frac{OwnW}{4} + \frac{OvtW}{2}}{(1 + e^{-K(d_{safe}+\xi)})} \tag{2.47}$$

Remembering that the EgoVehicle should already be moved towards the left of the EgoLane and the safety requirement of [44], we can rewrite Equation 2.47 to

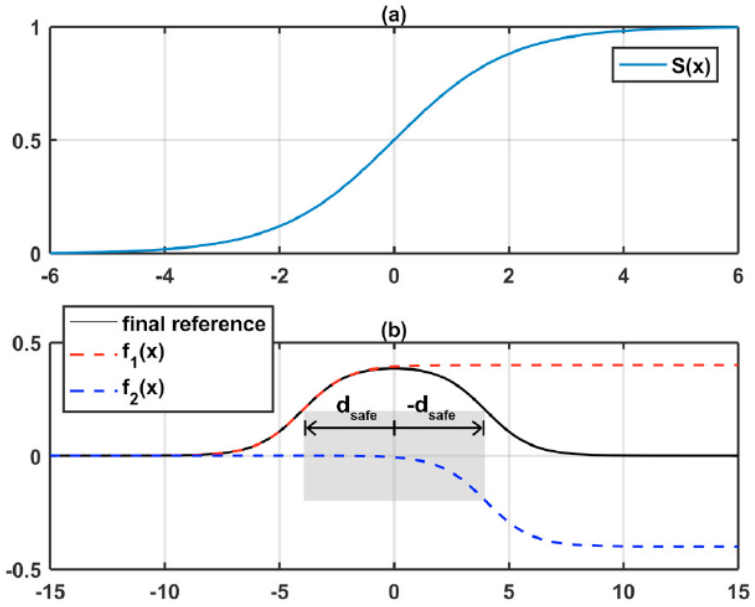

Figure 2.35: "(a) Sigmoid function with variable $x \in (-\infty, +\infty)$. (b) Overtaking path (black) composed of y_1 (red) and y_2 (blue). $w = 0.4, d_{safe} = 4 and \mu = 1$". From Figure 3 of [44]

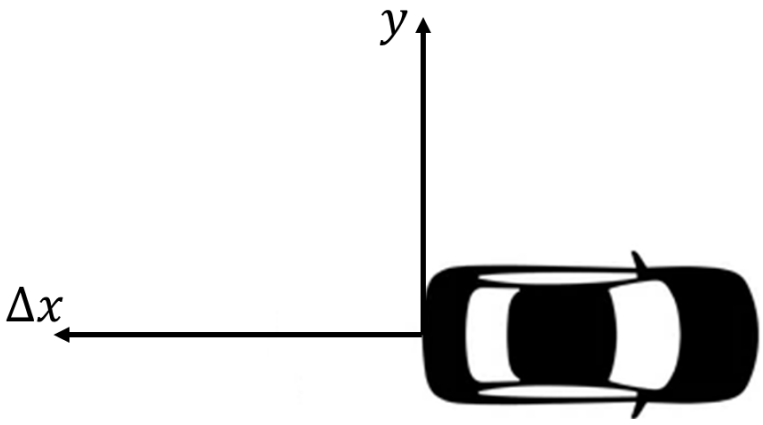

Figure 2.36: Reference Frame as we defined it

obtain ξ:

$$\xi = -d_{safe} + \frac{\ln\left(\frac{2OvtW}{OwnW}\right)}{-K} \tag{2.48}$$

where K is obtained by interpolation as in the Algorithm for Simplified Sigmoid and d_{safe} is obtained by multiplying the speed of the vehicle V by 1 second: this is

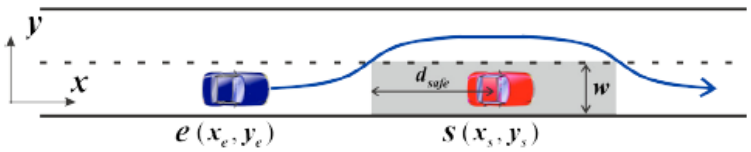

Figure 2.37: Figure 1 of [44], depicting the shape of the Sigmoid in relation to the safe distance

a large difference with respect to the solution of [44], which instead proposed

$$d_{safe} = \Delta V \cdot t_s \tag{2.49}$$

Such large difference is motivated, once again by our desire for safety, as the EgoVehicle should be working in an unmanned mode. Apart from this, the only difference lies in the fact that our Lane Change maneuver should span a narrower range, approximately 25% narrower, because of the fact we suppose to already be near the Boundary Line, while the safety constraint of being on said Boundary Line is kept immutated.
While we remain in the State 3 (OVERTAKE), the factors K and ξ are recomputed until the flag is dropped in a way similar to what done in Section 2.9.1, once the flag is dropped K and ξ are "frozen" and kept constant for the remainder of the Lane Change maneuver in the Overtake phase; in order to check if the flag is dropped, a test similar to the one displayed in the PseudoAlgorithm of Section 2.9.1 as in the following Equation

$$\begin{cases} y(\Delta x) = \dfrac{\dfrac{OwnW}{4} + \dfrac{OvtW}{2}}{1 + e^{-K(\Delta x + \xi)}} \\ y(\Delta x) > \dfrac{OwnW}{100} \end{cases} \tag{2.50}$$

The flag, as in Section 2.9.1, enables the Matlab function of the algorithm before, when it is not enabled, K and ξ are held constant and fed into a second block function, which corrects the e_{ct} and e_h which we receive from the Camera, to be sent to the Stanley Controller, as of Figure 2.38 One final remark about this local approach in opposition to the global approach of Section 2.9.1 is that, without the definition of a Global Path, we are able to perform it in real life, where it is difficult to have a completely straight road as was needed in that first step or an explicit formula describing the trajectory of the road, which would indeed be a very strict demand.
In order to prove such claim, we defined a dedicated Scenario, the Corner Overtake Scenario.

Algorithm 8 Final Sigmoid

function SIGMOID($V, x, OwnW, OvtW, K_{vec}, V_{vec}, \Delta X$) ▷ where K_{vec}, V_{vec} are the interpolants from Section 2.1
  $K = interp(V_{vec}, K_{vec}, V)$
  $d_{safe} = V \cdot 1s$
  Compute ξ from Equation 2.48
  if Equation 2.50 is true **then**
    flag = 0
  else
    flag = 1
  end if
  output K, ξ
end function

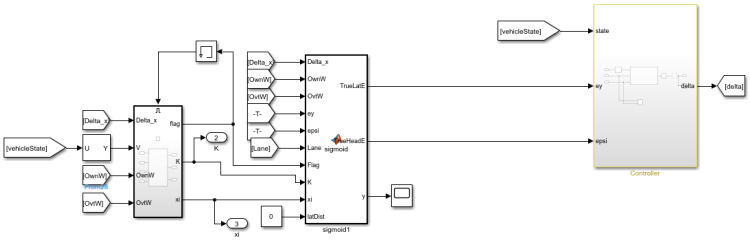

Figure 2.38: Simulink block from the Overtake section of our model

2.10 The Go Back Path Planning

Compared to the Overtake maneuver, the Go Back maneuver is simple: other than the fact that we can not rear end the leading vehicle - though we could still be rear ended ourselves in case the Leading/now Trailing Vehicle suddenly accelerates - we have also less troubles with the Lane Detection, since the presence of the lead vehicle can occlude the Camera vision of the Lane Markers, it is also easier because, without a leading vehicle, we can simply plan the Path to be followed as a function of x and not anymore as a function of ΔX.
This means that we can get rid of the range check of Equation 2.50 and define the Sigmoid Curve once and for all, to be followed immediately as soon as we enter into State 6 (GO BACK), since we have no leading vehicle as reference; this means that, instead of computing a new Sigmoid every time, with its K and ξ, we compute it just once, as we can see from the comparison between their Block Diagrams.
(Figures 2.38 and 2.39)
Furthermore, another difference is the variation of the K factor interpolants (see Section 2.1 for more details) and the fact that, given the fact that we employ the x position of the EgoVehicle instead of the ΔX from the leading vehicle (since there is no leading vehicle anymore), we need to integrate twice the longitudinal acceleration of the EgoVehicle (in a real application this could be performed with an IMU or an INS as in [30]) to obtain the **approximated** x value since we obtain the total length of the Path followed by the EgoVehicle, instead of its projection along the global X axis.
Apart from these differences, the procedure is the same, with the Sigmoid Curve simply reverted from Equation 2.47

$$y(x) = \frac{\frac{OwnW + OvtW}{2}}{1 + e^{K(x - d_{safe})}} \tag{2.51}$$

As we can see, the x at exponent is not preceded by the $-$ sign, meaning that the y value decreases going forward; it is also important to notice that the width of

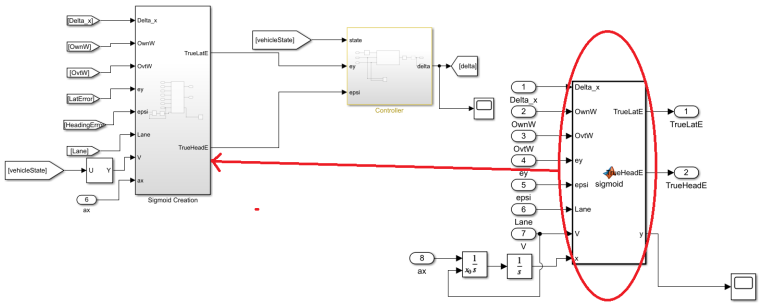

Figure 2.39: Simulink block from the Go Back section of our model

the Sigmoid is $\frac{OwnW + OvtW}{2}$ instead of $\frac{OwnW}{4} + \frac{OvtW}{2}$, this means that the Sigmoid Curve traces a path leading the egoVehicle to go back to the centerline of the OwnLane, instead of going back to the "waiting" position close to the boundary between Own Lane and Overtake Lane.

2.11 The State Machine

This Section is going to deal with the Decision Making performed by our System during its execution. The Decision Making is performed by mean of the control of the States of a Finite State Machine; the idea of using a State Machine, as an evolution of the first draft which was based on raising Flags called "overtaking", "going back" and so on, was inspired by [87], where the Deep Q-Learning Neural Network is moving in a so-called "action space" (Figure 2.40).

The main difference between our State Machine and the Action Space of [87] is

Number	1	2	3	4	5	6
Action	Change lane and Accelerate	Change lane and Keep speed	Change lane and Slow down	Keep lane and Accelerate	Keep lane and Keep speed	Keep lane and Slow down

Figure 2.40: "Definition of DQN action space" from Table II of [87]

that, while those Actions describe particular maneuvers, we chose to create our States with regard to particular conditions experienced by the EgoVehicle during the travel and the State itself will deal with the decision to accelerate or slow down. Below are our 7 States.

- **State 1: STAY.** Is the "standard state", where generally the FSM will converge.
- **State 2: WAIT.** Is a transition state, after we had to abort an overtake or before we can perform one.

- **State 3: OVERTAKE.** Is the state which draws the Overtake Path and controls its following.
- **State 4: EMERGENCY OVT.** Is the state where we move if the Ovt-Counter is triggered and it is not possible to abort any more.
- **State 5: ABORT.** Is the state where we move if the OvtCounter is triggered while we begin the Overtake.
- **State 6: GO BACK.** Is the state drawing the Go Back path and ensuring that we follow it.
- **State 7: BRAKE.** Is the emergency state, calling for a sharp brake. It is also the default State (for safety reasons).

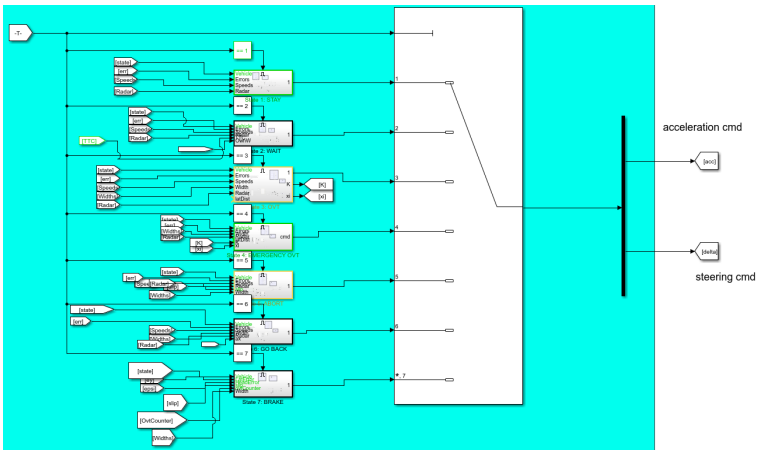

Figure 2.41: The scheme of our State Machine, with the Multiport Switch selecting the correct commands

Figure 2.41 shows the Simulink implementation of the State Machine: each State has its corresponding **enabled** block, which outputs a vector of commands $[acc,\ \delta]$: the Multiport Switch selects the correct vector according to the selected State. We use the FSM to juggle between the different states which, in turn, use different controls among the ones presented before: below, we are going to present the States one by one, each with its controls and the possible Next State.

2.11.1 State 1: Stay

As said in the list of all the States, this is the State to which the EgoVehicle will generally converge. Is the State in which our vehicle stays in the OwnLane, as said

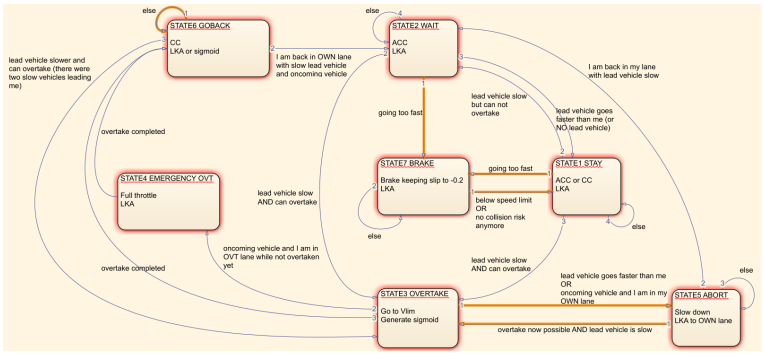

Figure 2.42: A simplified version of the Finite State Machine relations

by the name itself, be there a leading vehicle or not; in case there is no leading vehicle, the longitudinal control is the traditional Cruise Control, which has, as target, the lowest value between the $VelDes$ (target speed set by the user) and 95% of the Limit Speed, as said in Section 2.8.1; if, on the contrary there is a leading vehicle which is keeping a reasonable speed, so that our Vehicle does not lose too much time in following it, the longitudinal control is instead the ACC, in order to avoid rearending accidents with the lead vehicle. The selection between these two options is based on a Switch, which is enabled by the following proposition:

$$\begin{cases} Leading \wedge Followable \\ \\ where\ Followableis \\ \\ V_{lead} \leq min\Big(VelDes + 2,\ 0.95Vlim\Big) \end{cases} \tag{2.52}$$

which is translated into block diagram by Figure 2.43. Below are listed the parameters of the 1 Hz PID employed on the simple Cruise Control branch:

- **Proportional:** -10
- **Integral:** -2
- **Derivative:** 0
- **Filter coefficient:** 100

While the longitudinal control is more complicated, with different options in case of leading vehicle, too fast leading vehicle or being alone, the lateral control is much more simple, carried out by a base Stanley Controller as described in Section 2.8.3.

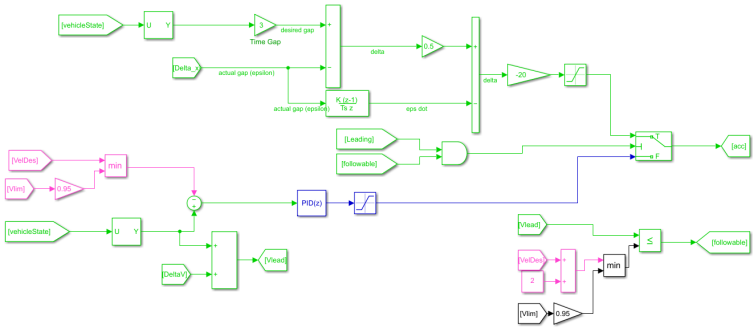

Figure 2.43: The Simulink block diagram representing the longitudinal Controls in State 1 (STAY)

Next State

The Next State is regulated by the Next State function, which - by mean of a switch function - selects the Next State according to the Current State and to the environment surrounding the EgoVehicle.
Since this is the same for all the 7 States, we wanted to give out this brief explanation, which will not be repeated for the latter States. In Figure 2.44 we can see the block diagram referred to the Next State selection, while the Matlab code itself will be discussed State by State. In case we are above the speed limit $Vlim$ or we are too close to the leading vehicle, we will want to move to **State 7 (BRAKE)**; in our system, we also had to add a $\&\&OvtCounter$ to the Time To Collision branch, because otherwise - in Scenarios where the EgoVehicle needs to overtake as soon as the Scenario starts - it would create problems due to fringe situations.
If we do not need to brake immediately, we need to check on the speed of the lead vehicle (computed in the block by mean of EgoVehicle speed and radar signal ΔV: if the lead vehicle is too slow for us, we will move into one out of 2 possible states, according to whether an Overtake maneuver is possible; if so, we will move to **State 3 (OVERTAKE)**, otherwise we will move to **State 2 (WAIT)**.
However, it is not only polite, but also **compulsory** to occupy the rightmost Lane [88] comma 5, so we would like to avoid being unnecessarily in the Overtake Lane. Because of this, a last check we perform is related to the Lane Number: if it is higher than 1 (Figure 2.29), we want to enter into State 6 (GO BACK). If none of the above checks are true, the EgoVehicle remains in State 1 (STAY).

2.11.2 State 2: Wait

This State is the one where the EgoVehicle generlly is when we **WANT** to overtake the lead vehicle, but we actually **CAN'T** overtake because of the environment

Algorithm 9 Next State for State 1 (STAY)

function NEXTSTATE(State, OvtCounter, OvtCompleted, Lane, Vlim, $\Delta V, \Delta x$, VelDes, e_{ct}, e_h, OwnW)
Retrieve VEgo
$TTC = -\dfrac{\Delta X}{\Delta V}$
$Vlead = VEgo + \Delta V$
if $VEgo > Vlim || (TTC < 1.5 \&\& \ OvtCounter)$ **then**
NextState = 7
else if $Vlead < min\Big(VelDes,\ Vlim\Big) - 2$ **then**
if O **then**vtCounter
NextState = 2
else
NextState = 3
end if
else if $Lane > 1$ **then**
NextState = 6
else
NextState = 1
end if
Output NextState
end function

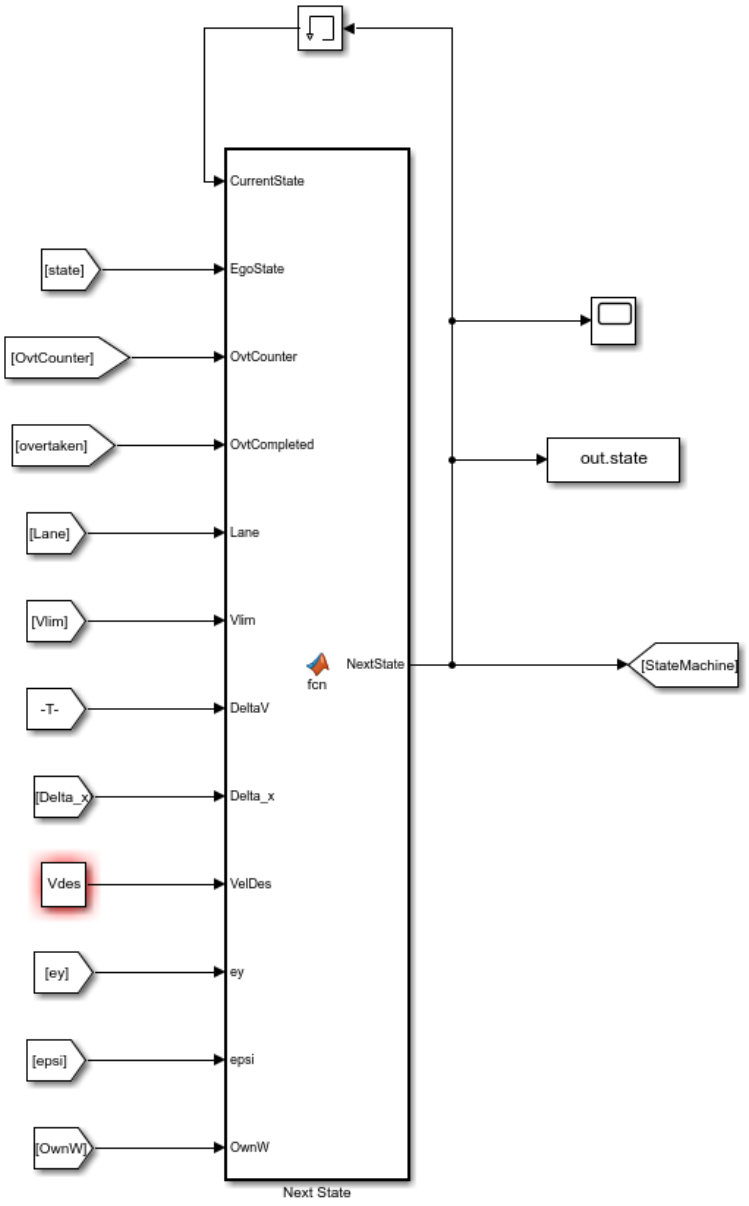

Figure 2.44: Block diagram of the Next State selector

around us (e.g. Solid Lane Marker or oncoming vehicles). Since this State is intrinsically linked to the presence of a lead vehicle, the Longitudinal Control is of course the Adaptive Cruise Control with the same gains as the one employed in State 1 (STAY), which are the ones described in Section 2.8.2: while in the State 1 (STAY) we had to first of all check that the lead vehicle is followable (Figure 2.43, **the fact that we are in State 2 implies that the lead vehicle is slow**

indeed, as we want to overtake it. In case the lead vehicle suddenly accelerates and goes above our desired speed $Veldes$ or the speed limit $Vlim$, the Next State will make the State move immediately to State 1 (STAY).
On the other hand, the Lateral Control sees a change from the "base State", which is State 1 (STAY): the Cross-Track Error e_{ct} is in fact corrected by a percentage of the LaneWidth as in Equation 2.53

$$e_{ct,corr} = e_{ct} + 0.25 OwnWidth \tag{2.53}$$

It is important also to note that the e_{ct} used in this Equation follows the **coordinate system of our Block Diagram, i.e. opposite to the "correct" one**, due to how the errors are computed from the coordinates of the Lanes in Camera data analysis (Section 2.7.4): this means that a positive e_{ct} in the Simulink block corresponds to a real negative e_{ct} and as such will be presented in the Results Section, Section 3; in this way, the EgoVehicle will be stable with a Cross-Track Error equal to $-0.25 OwnWidth$, i.e. $+0.25 OwnWidth$ in the Real Coordinate System (Figure 2.33). With a LaneWidth of 3 m, the e_{ct} will be -0.75 m, which means that the Left Boundary will be placed at $+0.75m$, while the Right Boundary will be located at $-2.25m$.

$$e_{ct} = \frac{LeftOffset + RightOffset}{2} = \frac{0.75 - 2.25}{2} = \textbf{-0.75} \tag{2.54}$$

The reason for this choice lies in the need to have an open unoccluded view of the Oncoming Lane in order to spot oncoming vehicles and at the same time to be already close to the Overtake Lane, so that the Overtake maneuver is faster.

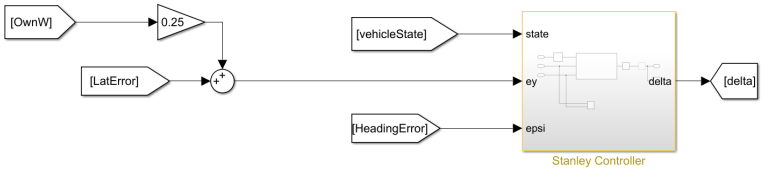

Figure 2.45: Block diagram of the Lateral Controller with the correction to force the EgoVehicle to stay close to the LaneMarker

Next State

The previously mentioned Next State function is used to compute the NextState also in case we are in the State 2 (WAIT), as is for every single State. Once again, as was the case for State 1 (STAY), the first check we perform is Safety-related; in fact, we first of all check the Time To Collision from the LeadVehicle as well as the speed limit $Vlim$: if we are not in safe conditions, we move immediately to **State**

7 (BRAKE).
Otherwise, we then check if the desired Overtake is now possible as well as necessary, in which case we move to **State 3 (OVERTAKE)**; the last check is on the followability of the lead vehicle, as we said at the beginning of Section 2.11.2: if the lead vehicle suddenly accelerates or disappears (because for example the lead vehicle turned off the road), we move back to the State 1 (WAIT); this is also needed as we will fall into State 2 (WAIT) when exiting of the State 6 (GO BACK) and there will probably not be a lead vehicle after the Overtake.
We also want to highlight that we removed, from this State's NextState determination, the possibility to move into the rightmost Lane in case we are not in it: while it is compulsory by law to occupy the rightmost Lane, in this case we would be faster than the lead vehicle, since - once we move to the rightmost Lane - the Lead vehicle would not be in range anymore and we would accelerate to our target speed $VelDes$: **this would lead to us overtaking on the right the slower lead vehicle, which is extremely dangerous and much more prohibited than occupying the Overtake Lane in the highway when the rightmost Lane is available.**

Algorithm 10 Next State for State 2 (WAIT)

function NEXTSTATE(State, OvtCounter, OvtCompleted, Lane, Vlim, $\Delta V, \Delta x$, VelDes, e_{ct}, e_h, OwnW)
  Retrieve VEgo
  $TTC = -\dfrac{\Delta X}{\Delta V}$
  $Vlead = VEgo + \Delta V$
  if $VEgo > Vlim$ || ($TTC < 1.5$ && $OvtCounter$) **then**
    NextState = 7
  else if $OvtCounter == 0$ & & $\Big(VelDes,\ Vlim\Big) - 2$ **then**
    NextState = 3
  else if $Vlead > VelDes - 2$ || $isnan(Vlead)$ **then**
    NextState = 1
  else
    NextState = 2
  end if
  Output NextState
end function

2.11.3 State 3: Overtake

This State is by far the most important of the 7 States we are discussing as well as the most complex in terms of Controls, as the State 3 (OVERTAKE) is the "core" of the whole System, controlling the EgoVehicle in creating the Sigmoid and performing the Overtake.
We begin by commenting the Longitudinal Controls, which is a simple Cruise Control, as we do not need to accelerate to the Speed Limit $Vlim$ in order to perform the Overtake, as the lead vehicle is traveling considerably below the desired speed. Therefore, the Longitudinal Control is basically the same as the one depicted in Figure 2.43, with the elimination of the **Followable** check and the ACC removed, as of Figure 2.46.

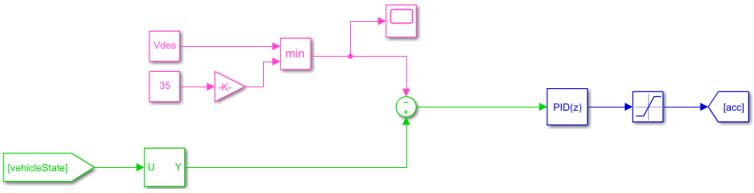

Figure 2.46: Block diagram of the Longitudinal Controller for the Overtake State

The Overtake: Sigmoid Creation

The most complicated part of the Controller for what concerns the State 3 (OVERTAKE), however is the Lateral Control, given that the EgoVehicle has to perform a Lane Change and that is indeed challenging; in order to build a system as simple as possible, we wanted to enforce such Lane Change by mean of the usual Stanley Controller and we managed to do this by "correcting" the errors given by the Camera (Section 2.7.4) through the Sigmoid Curve (Section 2.9.2, in a further development of what was already commented in the Section relative to the State 2 (WAIT) Controls (Section 2.11.2). The Simulink block diagram in charge of this maneuver has alredy been displayed in Figure 2.38, but we want to report it here for the sake of clarity. As already said in Section 2.9.2, the first block, which is enabled by a Flag, is in charge of computing the factor K and the ξ translation value, which is then fed to the second block, which is always executed as well as out of the State 3 (OVERTAKE) block. Once the EgoVehicle is close enough to the lead vehicle to have a Sigmoid Curve value high enough as of Equation 2.50,

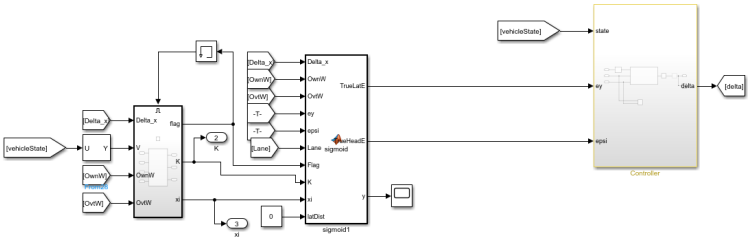

Figure 2.47: Simulink block from the Overtake section of our model

here recalled

$$\begin{cases} y(\Delta x) = \dfrac{\dfrac{OwnW}{4} + \dfrac{OvtW}{2}}{1 + e^{-K(\Delta x + \xi)}} \\ \\ y(\Delta x) > \dfrac{OwnW}{100} \end{cases} \tag{2.50}$$

the flag drops and K and ξ are freezed.
The core of this Lateral Control is in the second Matlab function, which is tasked with the correction of e_{ct} and e_h, which starts as soon as we enter into State 3 (OVERTAKE): in fact, just like we do in State 2 (WAIT), we want to move our EgoVehicle on the left border of the Lane, in order to have a fast Overtake maneuver. The full correction Algorithm is as follows.

Next State

The most important difference with the "Next State" from the previous 2 States we commented, STAY and WAIT, is that in no condition we move into **State 7 (BRAKE)**, since during the Overtake maneuver the speed limit is not the most concerning factor anymore, since we want to Overtake as fast as possible and the Speed difference is not so much dangerous since we will change Lane or we have done so already.
A check similar to the one leading to State 7 (BRAKE) is still in place, but leading to **State 5 (ABORT)**, from which we will eventually move into State 7 (BRAKE): this happens if the lead vehicle is faster than us and at the same time is close to our desired speed $VelDes$, since in this case we do not need anymore to overtake it.
If the lead vehicle has not increased its speed, the next check we perform, for Safety reasons once again, is on the value of OvtCounter: if the OvtCounter gets raised while we have not completed the Overtake yet, we then perform a basic check on what is safer, whether to abort the Overtake Maneuver and go back to the OwnLane - **State 5 (ABORT)** - or to push hard on the gas to complete the

Algorithm 11 Error Correction for Lane Change

function CORRECTION(Delta_x, OwnWidth, OvtWidth, ey, epsi, Lane, Flag, K, xi, latDist) ▷ where K, xi are given from the first block
  if Flag == 1 **then**
    $TrueLatE = ey + 0.25OwnWidth$
    $TrueHeadE = epsi$
  else
    Compute $y(Delta\_x)$ from Equation 2.50
    Compute $y\_next = y(Delta\_x - 1)$ from Equation 2.50
    $\theta = atan\left(\frac{y\_next - y}{-1}\right)$
    if $Lane == 1$ **then**
      $TrueLatE = ey + 0.25OwnWidth + y$
      $TrueHeadE = epsi - \theta$
    else
      $TrueLatE = ey + -0.5OwnWidth - 0.5OvtWidth + y$
      $TrueHeadE = epsi - \theta$
    end if
  end if
  if isnan(Delta_x) **then**
    $TrueLatE = ey$
    $TrueHeadE = epsi$
  end if
end function

Overtake as fast as possible - **State 4 (EMERGENCY OVT)** - and reenter in the OwnLane without the lead vehicle in front of us, **because if we went back to the OwnLane while still behind the lead vehicle we would risk rear ending it.**
If everything is fine and there are no Safety concerns, we then check the Overtaken Flag, in order to know if we can consider concluded the Overtake maneuver or not; it is also important to highlight the test condition on the branch moving to **State 6 (GO BACK)**: in fact we need to be in Lane 2 to start the Go Back maneuver from State 3, otherwise we would risk getting a Go Back maneuver triggered by every Vehicle parked on the side of the road.

Algorithm 12 Next State for State 3 (OVERTAKE)

function NEXTSTATE(State, OvtCounter, OvtCompleted, Lane, Vlim, $\Delta V, \Delta x$, VelDes, e_{ct}, e_h, OwnW)
  Retrieve VEgo
  $TTC = -\dfrac{\Delta X}{\Delta V}$
  $Vlead = VEgo + \Delta V$
  if $VEgo < Vlead$ & & $Vlead > VelDes - 2$ **then**
    NextState = 5
  else if $OvtCounter > 0$ & & $\Big(OvtCompleted == 0 || isnan(OvtCompleted)\Big)$ **then**
    if $Lane == 1 \&\& TTC > 2$ **then**
      NextState = 5
    else
      NextState = 4
    end if
  else if $OvtCompleted > 0$ & & $Lane == 2$ & & $isnan(Vlead)$ **then**
    NextState = 6
  else
    NextState = 3
  end if
  Output NextState
end function

2.11.4 State 4: Emergency Overtake

This State is a particular modified version of the State 3 (OVERTAKE) in which we enter in case we need to complete an Overtake maneuver as fast as possible, because we can not back down. It is to be noted, from Figure 2.48 that in this

State we do not have two Matlab function blocks as in the regular Overtake block diagram (Figure 2.38): this is because, if we are in such condition, we are in the Overtake Lane or anyway we have at least passed the distance at which K and ξ are freezed: because of this, the two values are coming from outside of the State 4 (EMERGENCY OVT) block, notably from the block 3, which is now not enabled and therefore is holding the values for K and ξ, so we only need the second part of the Error Correction block series.

We must note - however - that for the Lateral Control this block is exactly the

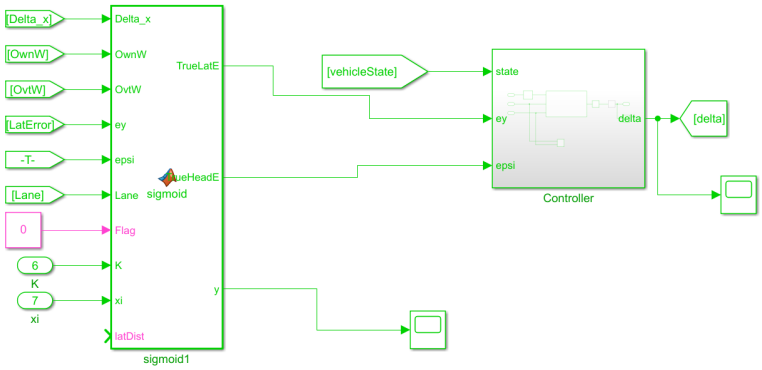

Figure 2.48: Simulink block from the Emergency Overtake section of our model

same as what we would have got with the base State 3 (OVERTAKE) Control block; what is different is the fact that the Longitudinal Control is not anymore performed by mean of a Cruise Control, but simply stating that the acceleration command must be as high as possible, as we need to get out of the Oncoming Lane as fast as possible, since the OvtCounter is raised and therefore we are in a dangerous situation, either because of an oncoming vehicle or because of a Continuous Lane Marker, which is drawn on the road in situations of danger.

Next State

Among all the Next State cases, this is by far the simplest, as we can exit from this State only if we have completed the Overtake maneuver: in such case, we would be moving to **State 6 (GO BACK)**, while otherwise we will keep staying in **State 4 (EMERGENCY OVT)**

2.11.5 State 5: Abort

This State is the one that probably put us in the hardest condition: this is because, while in STAY, WAIT etc we are more or less in the same conditions every time, we can enter in the State 5 (ABORT) while being in many several different conditions, for example:

Algorithm 13 Next State for State 4 (EMERGENCY OVT)

```
function NextState(Overtaken)
    if Overtaken then
        NextState = 6
    else
        NextState = 4
    end if
    Output NextState
end function
```

- We can enter in this State while we are still very far from the lead vehicle, the Sigmoid Curve is still outputting very low y value - i.e. we do not need to start the Lane Change yet - and so in this case we only need to move to State 2 (WAIT) and it does not require much more.

- We can enter in this State while we are steering towards the left but still well in the Own Lane, just requiring to correct the Steering and go back to the position on the Lane Boundary.

- We can enter in this State while we are almost crossing the Lane Boundary, therefore requiring a heavy steering correction with just a small TTC, just high enough not to trigger the State 4 (EMERGENCY OVT)

Because of this great mix of possible conditions, tuning the controls and in general the **aim** of this particular State was not easy for us and in the end we settled for a condition which should, in our concept, lead to the EgoVehicle being in the "waiting position", i.e. on the Left Boundary of the Own Lane, especially given the "NextState Algorithm" for State 3 - the only State leading to State 5, as of Figure 2.42 - which leads to State 5 only in case we are still in the Own Lane: if we are already in the Oncoming Lane, in fact, we are sent into State 4 (EMERGENCY OVT) as there is no Vlead computed, given no signals from the Radar (as said in Section 2.7.5, the reason of which will be further clarified in Section 3.9). Because of this, the Lateral Control of the EgoVehicle in this State is taken from the State 2 (WAIT) with the same corrections and the same gains on Stanley Controller.
On the other side, since this is one of the 3 States in which we are likely to be braking quite heavily or to be in need to slow down anyway (5 - 6 - 7), we set the Longitudinal Controller to be keeping the front tires slip at around -0.2, which is the value giving the higher longitudinal force F_x, as well as the fact that we are applying braking torque on the rear axle as well, as stated in Section 2.8 and visible in Figure 2.13. Here are the parameters of the 10 Hz PID in charge of the Longitudinal Control:

- **Proportional:** -125

- **Integral:** -45

- **Derivative:** 0

- **Filter coefficient:** 100

It is to be highlighted that the higher coefficients compared to the previous PID (Section 2.11.1) is due to the fact that here the errors fed to the PID are errors on the slip and not on the speed, so orders of magnitude lower.

Next State

Since we have previously been in State 3 (OVERTAKE), the first check we perform, even before the usual Safety check on the Time To Collision is the one on the possibility of Overtaking: if it is **possible** and at the same time **we want to overtake**, the State moves back into **State 3 (OVERTAKE)**; otherwise, if we are in the OwnLane and we are close to the "waiting position", we move into **State 2 (WAIT)**. If we on the other hand are having troubles with the safety distance from the lead vehicle, we move into the **State 7 (BRAKE)**, while in every other case we remain in State 5 (ABORT).

Algorithm 14 Next State for State 5 (ABORT)

function NEXTSTATE(State, OvtCounter, OvtCompleted, Lane, Vlim, $\Delta V, \Delta x$, VelDes, e_{ct}, e_h, OwnWidth)
  Retrieve VEgo
  $TTC = -\dfrac{\Delta X}{\Delta V}$
  $Vlead = VEgo + \Delta V$
  if $OvtCounter == 0$ & & $Vlead < VelDes - 2$ **then**
    NextState = 3
  else if $Lane == 1$ & & $abs\Big(ey + 0.25 OwnWidth) < 0.01 OwnWidth\Big)$ **then**
    NextState = 2
  else if $TTC < 1.5$ **then**
    NextState = 7
  else
    NextState = 5
  end if
  Output NextState
end function

2.11.6 State 6: Go Back

This State is tasked with the Lateral and Longitudinal Controls needed in order to move the EgoVehicle back to the normal position we would expect, i.e. to the centerline of the OwnLane, once the Overtake has been completed.
For what concerns the Lateral Control, the principle is the same we applied for the Overtake (Section 2.11.3) and the Emergency Overtake (Section 2.11.4, given that all of these States require the EgoVehicle to perform a Lane Change; therefore, as shown in Figure 2.49, the block diagram is composed of the Sigmoid Creation, used to correct the Errors coming from the Camera, and the Stanley Controller, fed by the aforementioned Errors in order to follow the created Sigmoid.

The fact that there is only a single Sigmoid Block, instead of the two present in

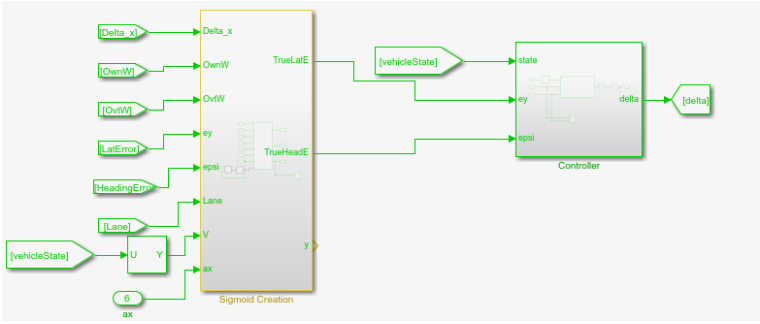

Figure 2.49: Simulink block from the Go Back section of our model

State 3 (OVERTAKE), is not due to the fact that K and ξ are freezed, as is in the State 4 (EMERGENCY OVT) block, but to the fact that we actually do not need to freeze them: since the Sigmoid is not a function of the relative distance anymore, but just of the longitudinal distance we ran, we can keep computing the y value during the run. In general, this State is enacting what was already pointed out in Section 2.10 for what concerns the Lateral Control.
Since we might have entered into Go Back State from both State 3 (OVERTAKE) and from the more aggressive State 4 (EMERGENCY OVT), we might be well in excess of the speed limit and our desired speed: for this reason, State 6 just like State 5 and 7 "simulates" the braking by applying the same negative torque to the rear wheels too (Figure 2.13). The PID is different from the ones employed in Section 2.11.1, which has the same working principle and is working at 10 Hz, below are the coefficients:

- **Proportional:** -20
- **Integral:** -4.5
- **Derivative:** 0
- **Filter coefficient:** 100

Next State

In order to understand why we do not account for the presence of a lead vehicle in this computation of the Next State, we must remember what was stated in Section 2.7.6: in fact, in order to raise the Overtaken Flag, the Central Radar must not be detecting any leading vehicle (as is seen in the Multiple Overtake Scenario in Section 3.9), therefore it is extremely unlikely that in the Go Back maneuver we detect such LEading Vehicle. The only way out of the State 6 (GO BACK) is to

State 2 (WAIT) in case the EgoVehicle is steadily back on the centerline of the Own Lane.

Algorithm 15 Next State for State 6 (GO BACK)

```
function NextState(Lane, ey, epsi)
    if Lane == 2 then
        NextState = 6
    else if abs(ey) < 0.01 & & abs(epsi) < 0.1 then
        NextState = 2
    else
        NextState = 2
    end if
    Output NextState
end function
```

2.11.7 State 7: Brake

This State is the last of the 7 States that we defined for our Decision Making State Machine, as well as the Default State selected by the Multiport Switch (Figure 2.41) in case of failure; this was chosen by us because we wanted to be as safe as possible and, in case of failure, we prefer to be locked into a permanent braking maneuver rather than into an overtake maneuver or even worse into an emergency overtake, which has the acc command locked to 100 (Section 2.11.4).
Regarding the Longitudinal Controls, the Control we selected follows the same working principle as the State 5 (ABORT), i.e. a PID trying to keep the slip of the front axle as close as possible to -0.2, with the same torque applied to Front and Rear wheels (Figure 2.13), with the following coefficients:

- **Proportional:** -20
- **Integral:** -4.5
- **Derivative:** 0
- **Filter coefficient:** 100

For what concerns the Lateral Controls, the selected System is the simple Stanley Controller, without any Sigmoid-based correction, as presented in Section 2.8.3.

Next State

Because of the severe Safety issues correlated to this State, there is only a way out of this State, i.e. in case the need for Braking is exhausted: to fulfill this

requirement, we need to be at the same time well below the speed limit $Vlim$ as well as with a significant gap from the lead vehicle, meaning that the risk of rear-ending has passed. In this case, we will move to **State 2 (WAIT).**

Algorithm 16 Next State for State 7 (WAIT)

function NEXTSTATE(State, OvtCounter, OvtCompleted, Lane, Vlim, $\Delta V, \Delta x$, VelDes, e_{ct}, e_h, OwnWidth)
  Retrieve VEgo
  $TTC = -\dfrac{\Delta X}{\Delta V}$
  $Vlead = VEgo + \Delta V$
  if $VEgo < 0.85Vlim$ & & $\Big(TTC > 5$ || $isnan(TTC)\Big)$ **then**
    NextState = 2
  else
    NextState = 7
  end if
  Output NextState
end function

Chapter 3

Experimental Results

In this section, we will present and discuss the graphs with the results, Scenario by Scenario, in order to show that the Controls have been tuned to work and the State Machine we have defined in Section 2.11 is working.

3.1 ACC_Scenario

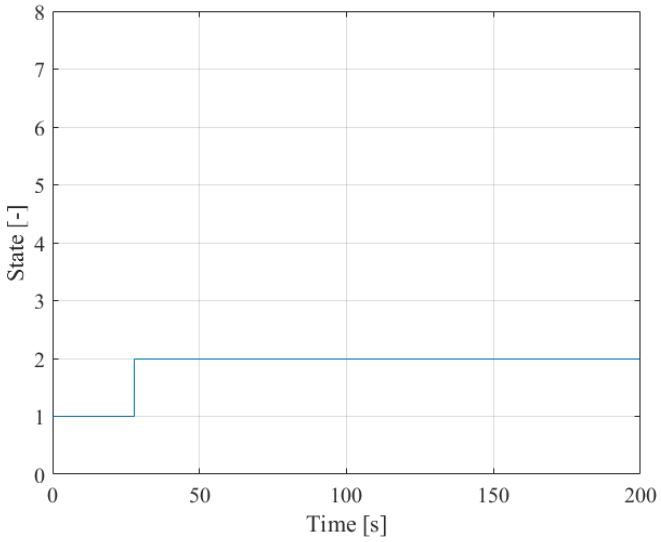

Figure 3.1: State Machine during the ACC_Scenario run

By looking at Figure 3.1, we can see that the State is constantly 1 (STAY), until around the 28 second mark, when the EgoVehicle arrives in the range of the leading vehicle, as is also confirmed by Figure 3.2 where the **Leading** flag is raised

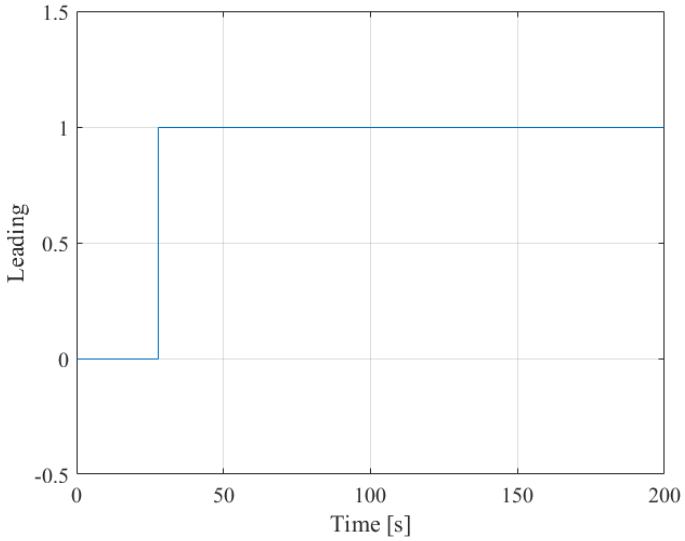

Figure 3.2: Leading Flag during the ACC_Scenario run

around the same time. Once the flag is raised, the State passes into 2 (WAIT), coherently with what we stated in Section 2.11.1.
Because of what we decided for the WAIT State, as soon as we enter into such State, the EgoVehicle begins a quick turn to the left, in order to move close to the Centerline, so that it is ready to perform an Overtake and - at the same time - its Left Side Radar is not occluded by the Lead Vehicle, so we can more easily spot an eventual oncoming vehicle.

The drop in speed at the beginning of the simulation is due to the fact that the initial value for the acceleration command is 0, as stated in Section 2.4.1; meanwhile, in a rather counterintuitive way, the entrance of the EgoVehicle into the WAIT State marks a sharp increase of the speed (Figure 3.3): the reason for this lies in the fact that the ACC is not a simple Adaptive Cruise Control aiming to keep a constant distance between us and the leading vehicle, but rather a Constant Time Gap, which aims to keep a gap of 3 seconds between the two vehicles, i.e. a variable distance between the two vehicles, equal to 3 times our speed, as defined in Section 2.8.2. This space is much larger than the usual 1 second [84][85] which is dictated by the human time of reaction: this is because, since our EgoVehicle is unmanned, we cannot allow to have such a small safety distance, because errors or failures in the radar might be unresponded by a Driver just acting as a passenger. Moreover, the "strange" acceleration is also due to the fact that we start in a fringe situation, because, if the simulation started with the car already in range of the Lead Vehicle, the speed profile would simply replicate the one of the Lead Vehicle. It is to be noted, however, that the distance never goes below the 3 seconds which

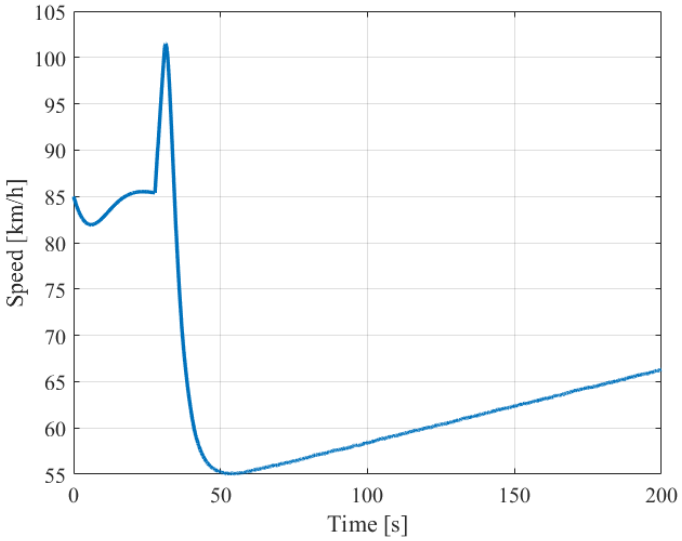

Figure 3.3: Speed profile of the EgoVehicle during the ACC_Scenario run

are our goal, as of Figure 3.4 and Figure 3.5. We can see from Figure 3.5 that

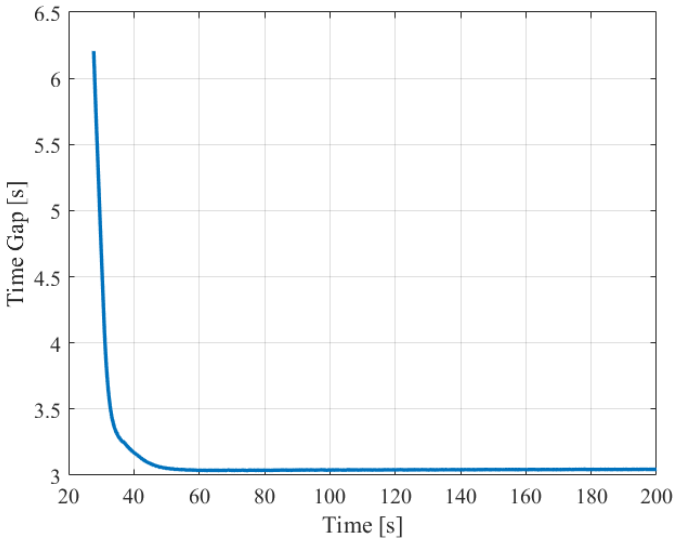

Figure 3.4: Time gap during the ACC_Scenario run

the signals coming from the radar are a bit noisy, but this can be attributed to the noise of the GNN tracker and the noise of the Radar block and moreover the amount of variation is negligible with respect to the value. The x-axis of Figure 3.5 does not begin at 0, of course, but around the 28 s mark: this is because the

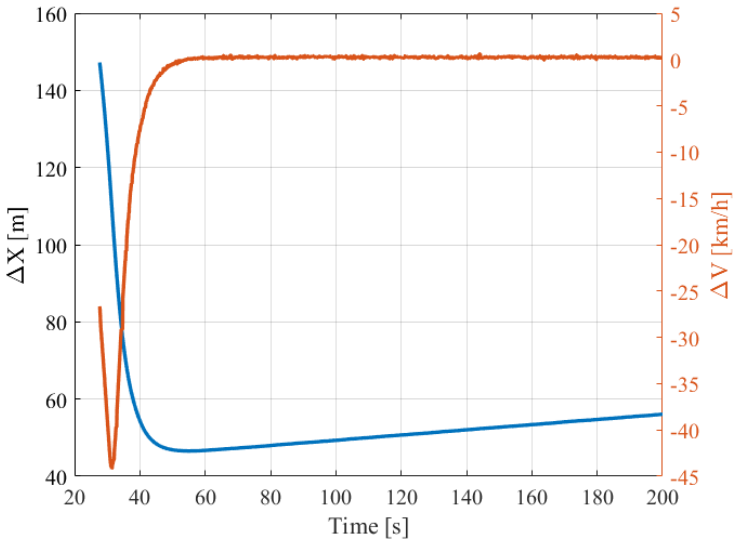

Figure 3.5: Radar signals during the ACC_Scenario run

leading vehicle is not in range of the central radar until this moment, as is also confirmed by Figure 3.2; coherent with this and with what was stated in Section 2.6, the first value of the ΔX signal is ≈ 150 m, which is the maximum range of the central radar. In the first 20 s of the STAY state, the ΔX value falls very quickly, because of the fact that the ΔV value is experiencing a negative peak, which is caused by the sharp peak in the EgoVehicle speed around the 30 s of Figure 3.3: in fact, we are much faster than the leading vehicle (compare with the data given out in Section 2.4.1), but we need to put ourselves at a distance of 3 s as fast as possible; in fact, after about half a minute, the relative velocity between us and the leading vehicle is circa zero, while the relative distance is increasing slowly but steadily, due to the fact that the leading vehicle is accelerating up to 72 km/h (20 m/s) and in fact the distance is increasing up to 60 m.

We also can assume that the data analysis functions for the radars' signals are working as expected, given that the Oncoming flag (Figure 3.6 has the trend that we would expect.

3.1.1 Second run with LKA active

We later performed a second simulation for the very same Scenario, this time with also the steering activated, i.e. the whole LKA system was included, as well as all the Path Planning blocks which were commented out during the first run; as we can see from Figure 3.1, when the Leading Car is in range of the radar, the

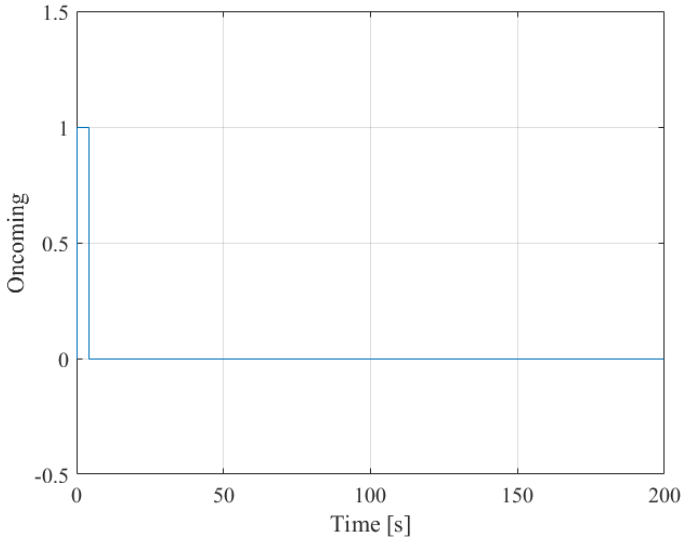

Figure 3.6: Oncoming flag during the ACC_Scenario run

State passes from STAY to WAIT: because of this, as now the whole system was activated, the Path Planning for the WAIT state entered into action, in order to move the car closer to the center. This is seen in Figure 3.7 The other main metric

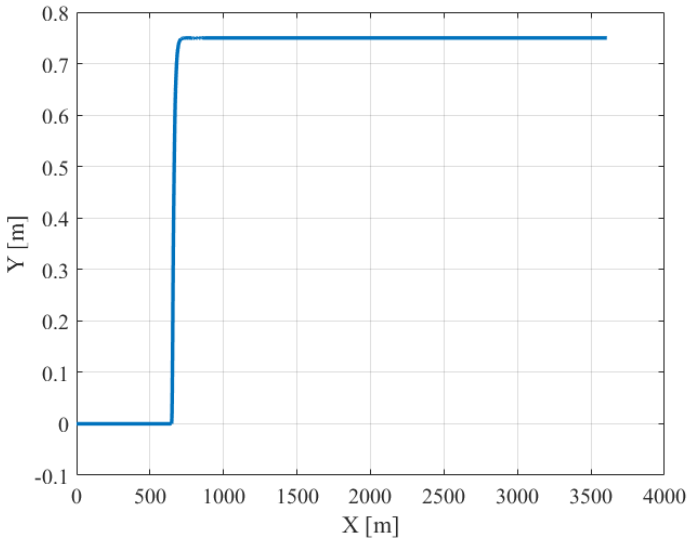

Figure 3.7: Trajectory followed during the **second** ACC_Scenario run

that we need to check is the Heading Error that the car experiences during the whole simulation (Figure 3.8) The Heading Error e_h has a sudden spike, which

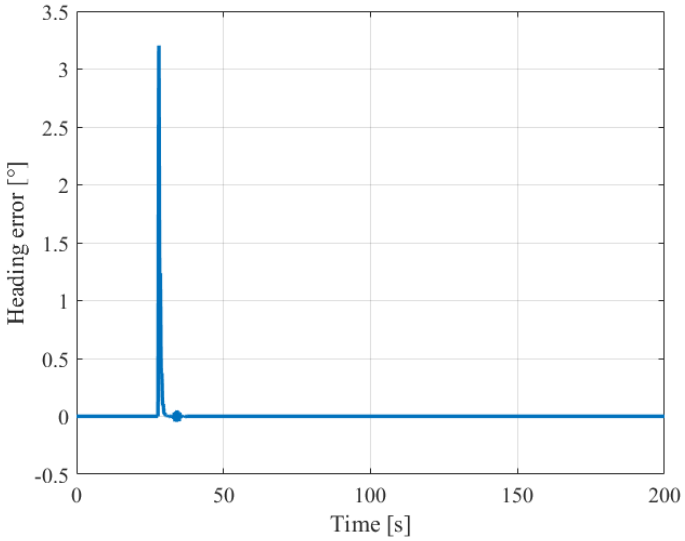

Figure 3.8: Heading Error e_h during the **second** ACC_Scenario run

is nevertheless very moderate (just 3ř) in correspondence of the initial steering command (Figure 3.9) input as soon as the EgoVehicle moves into the WAIT state, in order to move to the centerline; after this spike, it rapidly converges back to 0, except for a small oscillation of less than 0.1ř at the end of the maneuver.

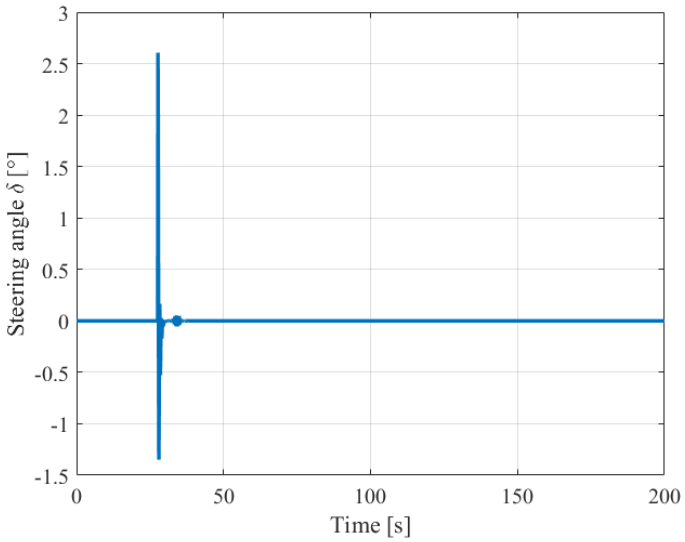

Figure 3.9: Steering Command δ during the **second** ACC_Scenario run

3.2 Skidpad

The first results we are going to show for what concerns this Scenario are the most obvious ones, which will require little discussion: since the Scenario is totally

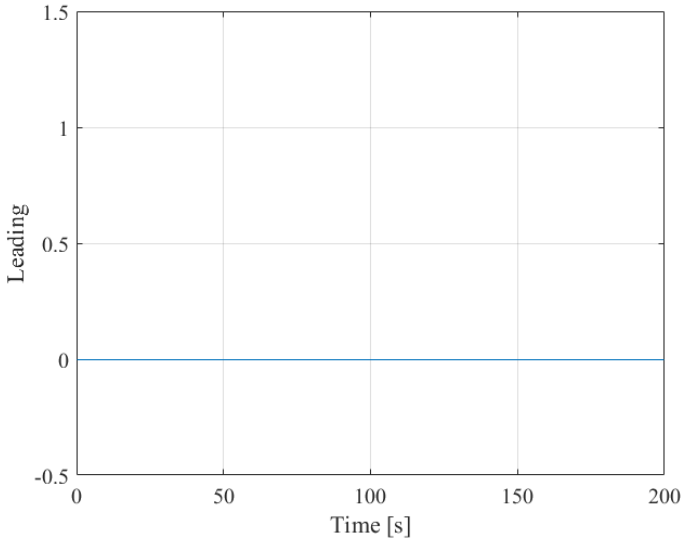

Figure 3.10: Leading flag during the Skidpad run

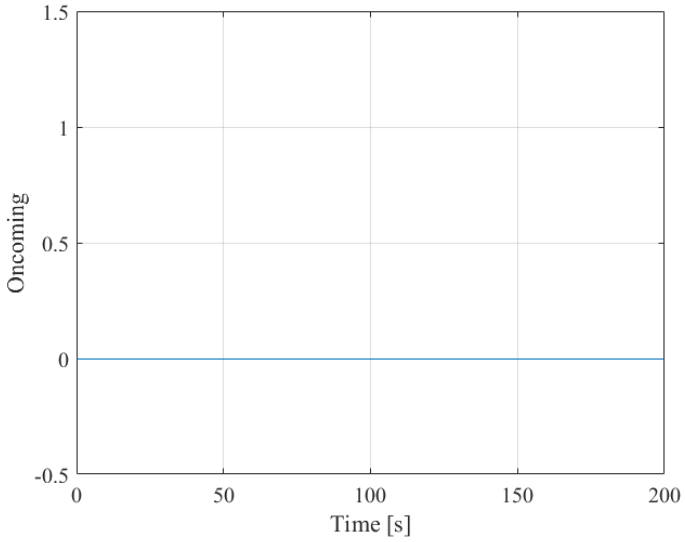

Figure 3.11: Oncoming flag during the Skidpad run

empty (Section 2.4.2), we expect that the Leading and Oncoming flags are never

raised; as we can see from Figure 3.10 and Figure 3.11, this is the case, confirming that the GNN tracker, which we discussed in Section 2.7, is working well to filter out noise detections.
Because of this, the radar signals are constantly NaN, as can be observed from Figure 3.12 which is completely empty leading to a NaN time gap as well (Figure

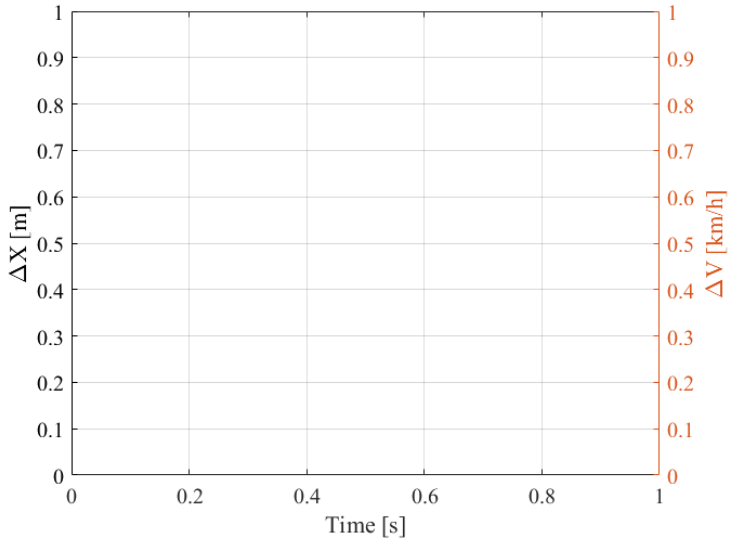

Figure 3.12: Radar signals during the Skidpad run

3.13). We presented these graphs which are completely empty and can seem meaningless just for this first Scenario where we have no other actors to testify that the data analysis on the Perception is working as expected and is not raising random Leading or Oncoming Flags or outputting random ΔX or ΔV signals. Once the most trivial results from this simulation have been discussed, we can move to the most concrete ones: we can for example see from Figure 3.14 that the simple Cruise Controller works well in a curve environment just like it did working in a straight (compare with the first 27 seconds of Figure 3.3) as after the first drop in speed (due to the fact that the initial acceleration command is 0), the speed rapidly converges to the value of $V_{des} = 85km/h$.
Moreover, as the Scenario was designed to test the steering performance, we conclude the Section by discussing the handling results, i.e. the Cross-Track Error e_{ct} and the Heading Error e_h as well as their effect, the steering angle δ as obtained from the **Stanley controller** discussed in Section 2.8.3; the first comment we can have is about the Cross-Track Error e_{ct}, which is negative: according to what we stated in Section 2.8.3, a negative e_{ct} means that the car is on the right of the centerline of the Lane, which we can attribute partly to the centrifugal acceleration caused by the high turning speed. We must in fact remember that the Vehicle

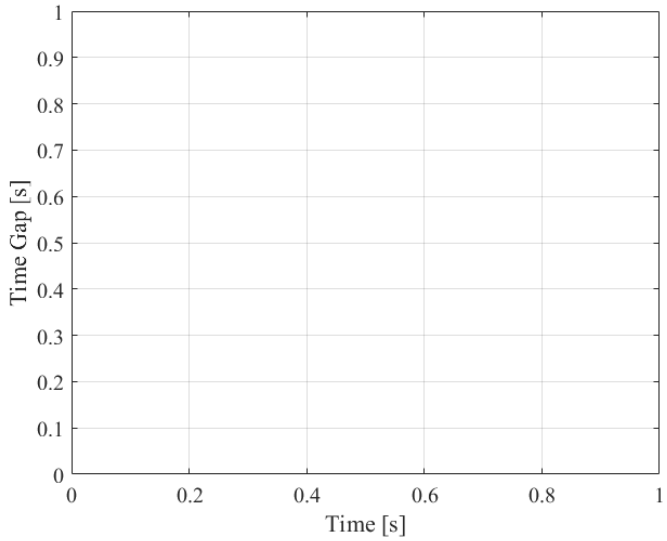

Figure 3.13: Time gap from the leading vehicle during the Skidpad run

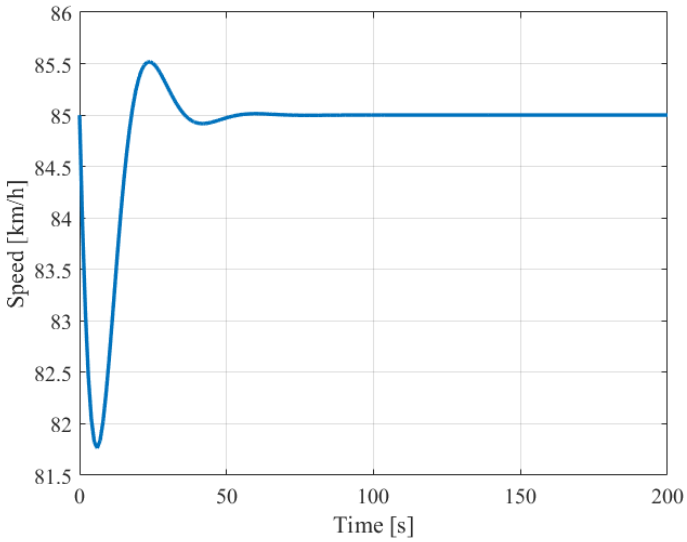

Figure 3.14: Speed of the EgoVehicle during the Skidpad run

Model used for these experiments is the Full Vehicle Model, therefore we must also account for the lateral acceleration and the local maximum (= minimum of the absolute value) for the e_{ct} happens right before the 10 s mark (Figure 3.15), around the same time where the vehicle has the minimum speed (Figure 3.14). Anyway, what we want to highlight is that such steady state e_{ct} is less than 20 cm, therefore

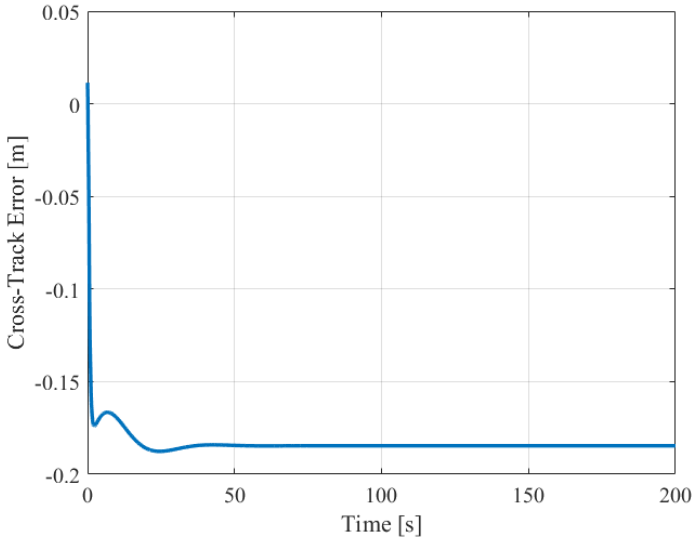

Figure 3.15: Cross-Track Error of the EgoVehicle during the Skidpad run

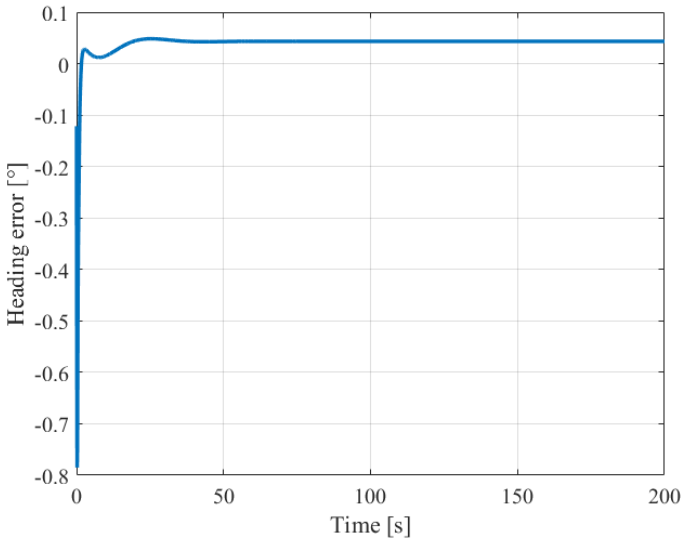

Figure 3.16: Heading Error of the EgoVehicle during the Skidpad run

being well in the range of acceptability, as we can see from Figure 3.18.

From the same Figure 3.18 we can see that the Heading Error e_ψ is almost negligible, as the Lanes (highlighted in red) are basically parallel to the car, coherently with the steady state error of 0.5° we see in Figure 3.16, which is a very low value, but something we can expect during a constant radius turn, let alone a turn with such

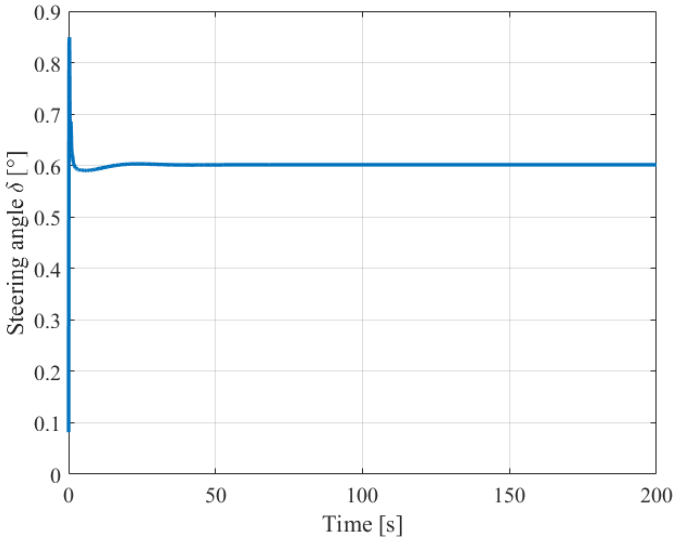

Figure 3.17: Steering Command during the Skidpad run

a large Turning Radius (300 m).
Finally, we can comment on the value of the steering angle δ: at first, it could look like the value of $\mathbf{0.6}^\circ$ is a mistake, as the maximum steering angle **on the ground**, for a common road vehicle can be $30^\circ - 35^\circ$; however, if we compute the kinematic steering angle δ_{kin} starting from the formula found in [58]

$$\frac{1}{R\delta} = \frac{1}{l} \tag{3.1}$$

reverted to

$$\delta_{kin} = \frac{l}{R} \tag{3.2}$$

knowing that the skidpad has a radius of 300 m, corrected to 300.19 m by the e_{ct} and the car's wheelbase l is 2.4 m yields out a kinematic steering angle of $\mathbf{0.458}^\circ$, which is even smaller than the one we are inputting; this is due to the fact that we are not in a kinematic condition, due to the high speed of our vehicle which leads to the arising of Aerodynamic Forces, which leads to an Understeering characteristic.
Moreover, if we compute the correction factor for the correct Curvature Gain, as of page *267* of [58]

$$\frac{1}{R\delta} = \frac{1}{l} \frac{1}{1 + \frac{m}{l^2}\left(\frac{b}{C_f} - \frac{a}{C_r}\right)\frac{V^2}{gl}} \tag{3.3}$$

when substituting the vehicle specifics - accounting also for the fact that the car is Front Wheel Drive and the front axle experiences the highest load transfer - gives

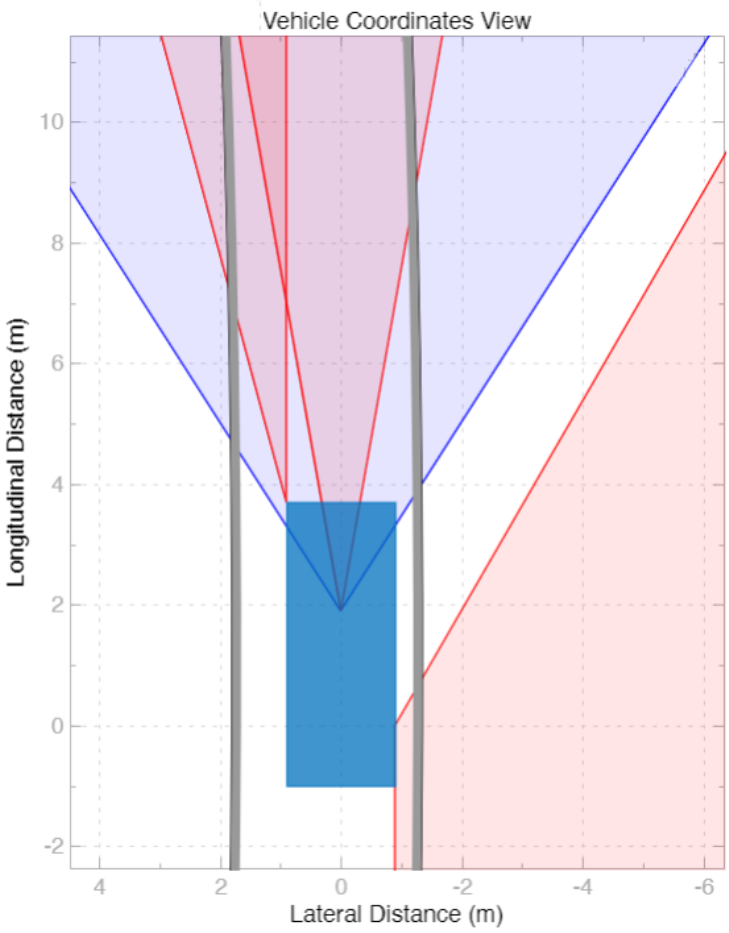

Figure 3.18: Screenshot from the Bird's-Eye Scope demonstrating that a steady state Cross-Track Error of $\approx 20cm$ is absolutely acceptable in a Skidpad

out a factor of around **0.76**, in agreement with the ratio between the kinematic steering angle and the dynamic steering angle. Such computations, for sake of simplicity, have been omitted.

3.3 Empty Turn Scenario

As for the Section 3.2, we begin with a quick comment of the results of the Radar block and the data analysis downstream in the pipeline:

the oncoming vehicle is correctly detected at the beginning of the Simulation: the flag, however, compared to the one of the ACC_Scenario (Figure 3.2) is up for a shorter period of time; the reason for this lies in the higher speed of the oncoming vehicle, 50% higher than the one in ACC_Scenario. Since there is no

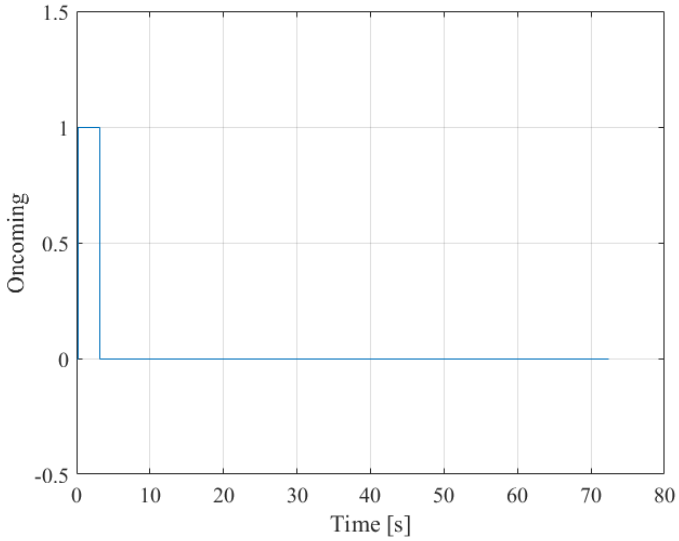

Figure 3.19: Oncoming flag during the Empty Turn Scenario run

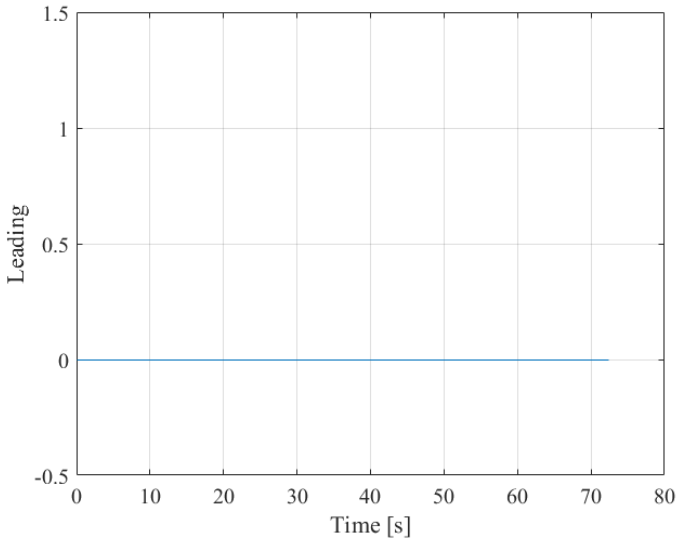

Figure 3.20: Leading flag during the Empty Turn Scenario run

leading vehicle throughout all the simulation, it is expected to not only see the Leading Flag constantly equal to 0, but also to have a completely empty graph regarding the Radar quantities, in parallel with what we commented already in Section 3.2 (Figure 3.10 and Figure 3.12), therefore confirming us that the GNN tracker is filtering well the noise and fake detections from the Radar.

Moving on the main goal of this Scenario, we introduce the results of the LKA: the main element which is noted is a strong spike in the value of the Cross-Track Error e_{ct} around the 15 s mark, with over a meter of error, leading to the car almost going off-road; the good thing is that this large spike is exclusive to e_{ct}, while e_h is still small (less than 5°, as of Figure 3.22) and it is in counteraction to the Cross-Track. (Peaks are around the 15 s mark as of Figure 3.23) We can see

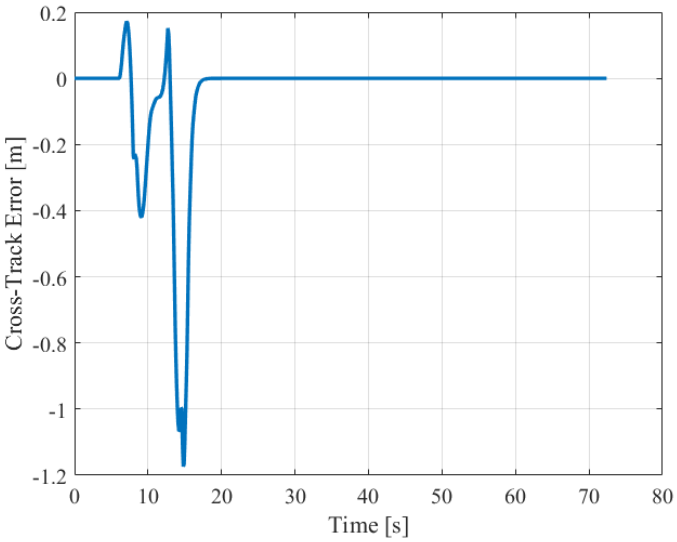

Figure 3.21: Cross-Track Error of the EgoVehicle during the Empty Turn Scenario run

that the high e_h is a consequence of the high e_{ct} by looking at Figure 3.23, where it is evident that the spike in the Cross-Track Error comes **before** the spike in the Heading Error; moreover, if we compare Figures 3.23 and 3.24, we can confirm the causal relation between the two spikes, as - in the time occurring between such spikes - is present the spike in steering angle, which is coherent also with the e_h, as the positive steering angle is correlated with a counter-clockwise rotation. At this point, this e_{ct} trend could look worrying, but it is not actually due to the instability of the Stanley controller, but rather to the difficulty of the Scenario, where the turn is not a constant radius turn, but has a very sharp exit with a sudden decrease in the radius (Figure 3.25) which caused this spike in e_{ct}, leading to the sudden steering angle leading to the spike in e_h. One final comment we can do is regarding the speed throughout the simulation, which is displayed in Figure 3.26: the initial drop in speed is due to the initial value of the acceleration command of 0, while the larger drop is due to the lateral slip of the vehicle during the sharp turn.

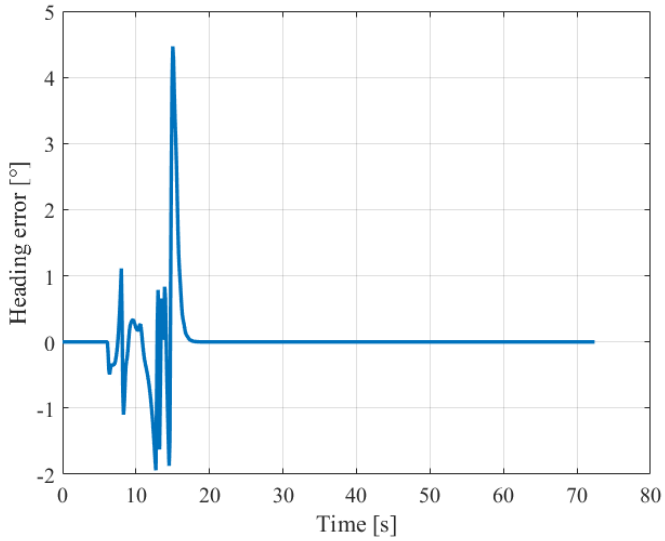

Figure 3.22: Heading Error of the EgoVehicle during the Empty Turn Scenario run

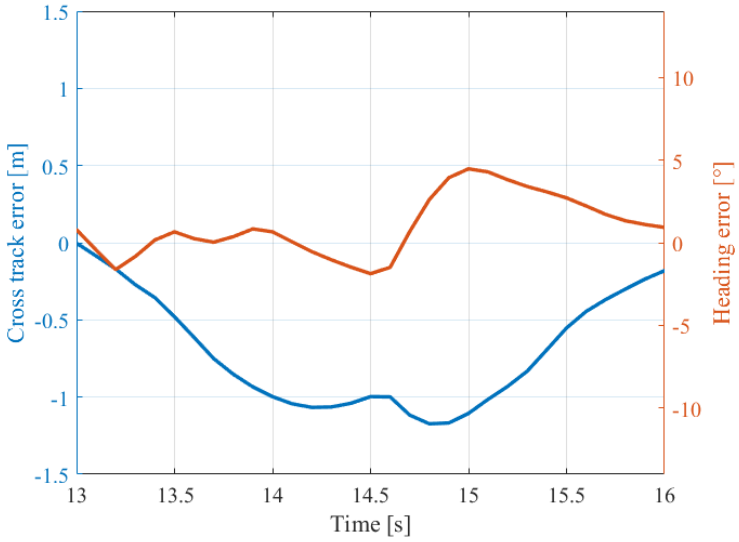

Figure 3.23: Zoom of the errors on the 15 s during the Empty Turn Scenario

3.4 Mountain Road

As for the Skidpad (Section 3.2) and the EmptyTurn (Section 3.3) Scenarios, we start off by commenting the "simplest" results, i.e. the Leading/Oncoming flags

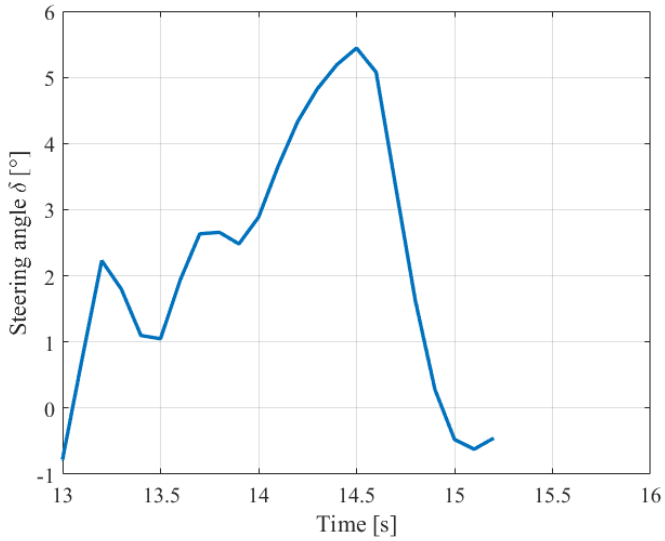

Figure 3.24: Zoom of the δ steering angle around the 15 s mark during the Empty Turn Scenario

- which are correctly always zero - and the radar signals, which are non existant given the absence of any other vehicle in the environment.

Figure 3.29 shows that the simple Cruise Control - as the EgoVehicle is alone - is able to keep the speed constant even throughout a challenging track, with only a dip below 50 km/h (due to the first sharp corner) and a subsequent overshoot which is still acceptable with a speed limit of 50 km/h [89].

The most important part about the results of this simulation comes from the comment on the lateral dynamics, i.e. the trend of the errors and of the steering command; Once again, similarly to what we have already noted in Section 3.3, there is a main spike in the e_{ct} as well as in the e_h trend, which happen in correspondence with the peak of the δ command; in this case the maximum value is large, 1.5 m would basically mean that the car went off track: the only good fact about this is that the value is -1.5 m (car going outside) instead of +1.5 m, which would mean that we invaded the Oncoming Lane.

Moreover, this value is only temporary, therefore our EgoVehicle manages to quickly recover the trajectory and get back in the Lane; we would also want to stress out once again the fact that this instability happens in correspondence of a particularly difficult corner, not with constant radius, where the Camera Lane Tracker experiences troubles in defining a clothoid to represent the Lane Boundaries [90].

This last remark is not intended to be a criticism towards the Simulink block, but rather an admission of the non perfect compliance of our Scenario to a real road; this is however, something we expected, since there are no standards, as of the

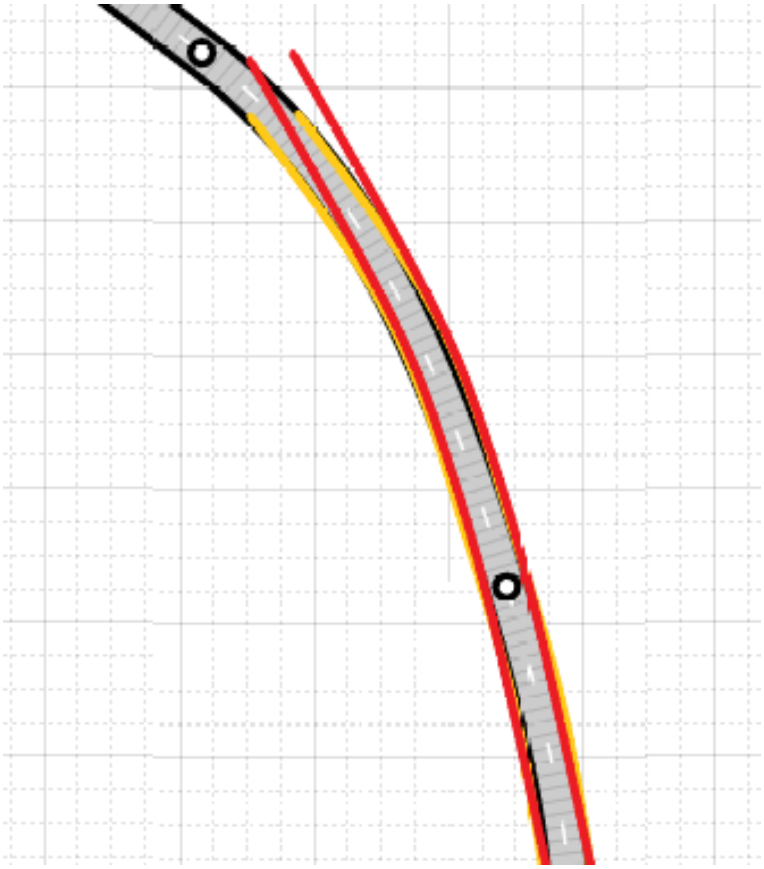

Figure 3.25: Zoom of the final part of the turn, in yellow the road limits, in red the track which would be correct

time of writing, regarding a Testing Scenario for an LKA system and this very same Scenario we presented was obtained after many trial-and error iterations of Scenarios which were either completely inadequate or too simple.
A final remark about this Scenario is a hint of what will be discussed in Section 4: the fact that the CC is able to keep a constant speed of 50 km/h is a good thing regarding the operation of the CC itself, but at the same time speaks volumes about the still simplistic approach we had for the speed profile: in case we do not have a leading vehicle, in fact, we assumed a Desired Speed V_{des} and a Limit Speed V_{lim} and simply took the lower of the two as our goal; as a future development, we would like to extend this logic to include a third limit value, i.e. the maximum speed that the curvature we are going to find allows us.
This would be possible since, as stated in [90], the Camera gives out the curvature of the Lane (hence the curvature of the road) in form of Clothoid and therefore it would be feasible to extract a profile of curvature and from it a profile of speed,

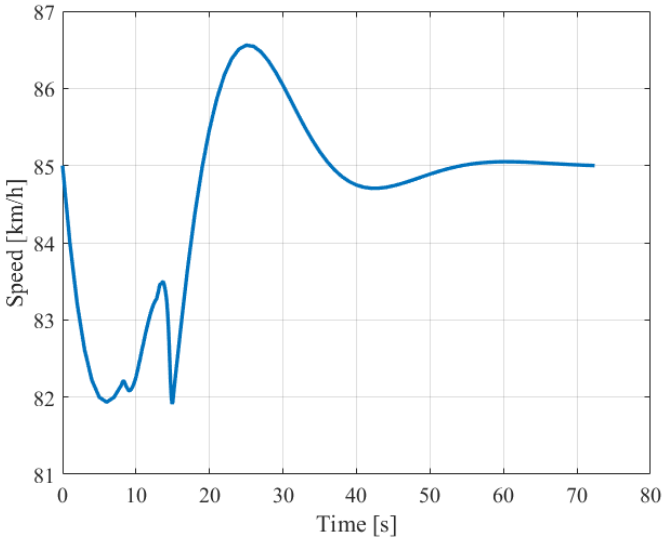

Figure 3.26: Speed of the EgoVehicle during the Empty Turn Scenario

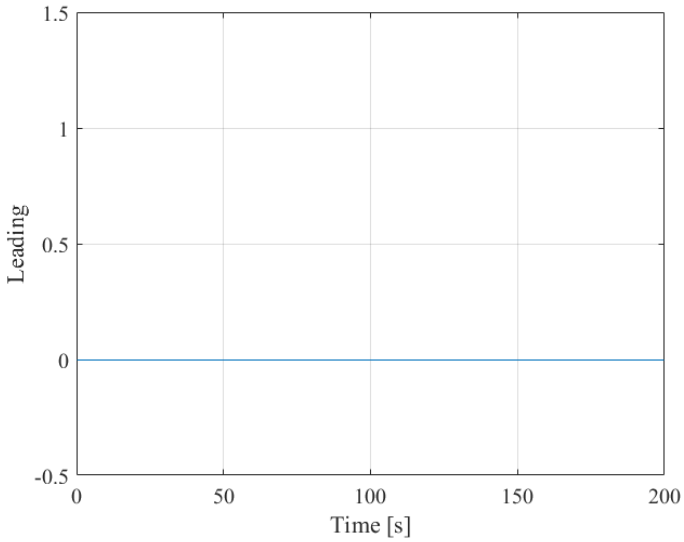

Figure 3.27: Leading Flag during the Mountain Road Scenario

following the ideas presented by another Squadra Corse DRIVERLESS member in [91].
Finally, the main focus of this thesis work lies in the Path Planning for the Overtaking, therefore such complex turning roads are not our main focus (so we did not discuss the possibility of a $Speed(curvature)$ function) and we wanted to introduce

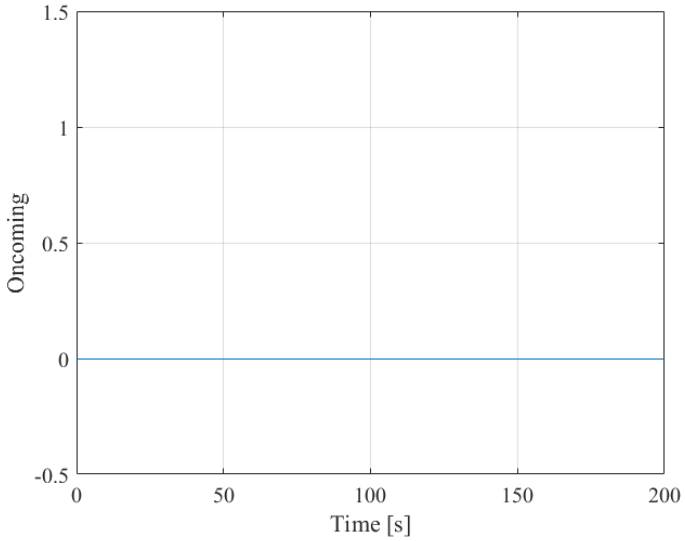

Figure 3.28: Oncoming Flag during the Mountain Road Scenario

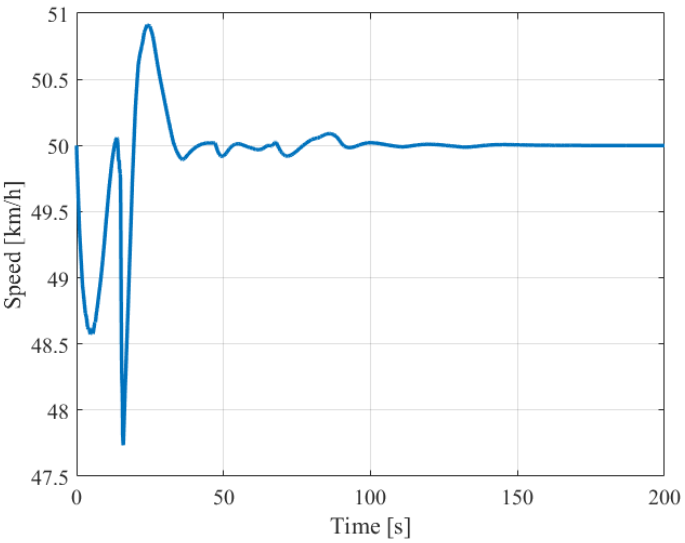

Figure 3.29: Speed during the Mountain Road Scenario

such Scenario just as a Final Test for our LKA.

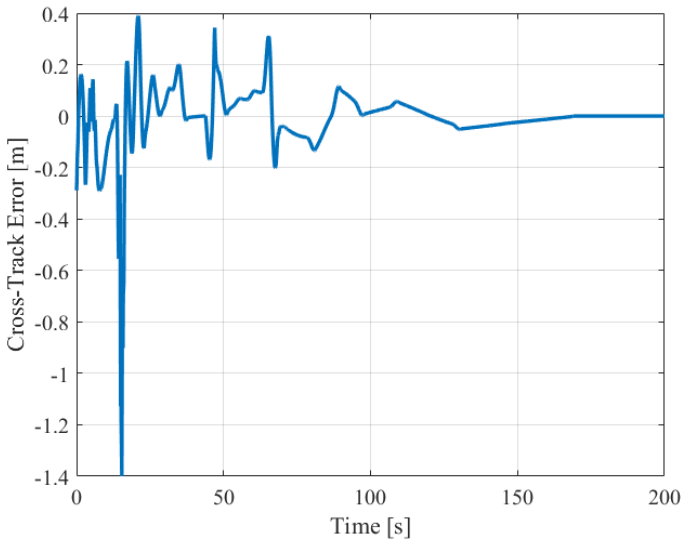

Figure 3.30: Cross-Track Error during the Mountain Road Scenario

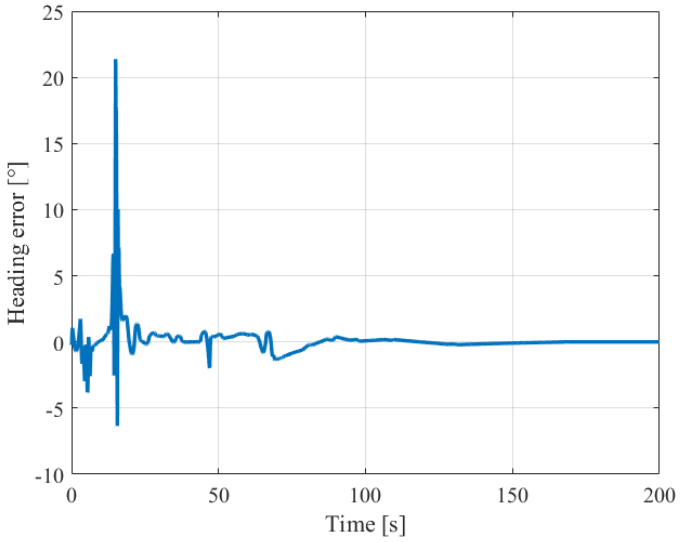

Figure 3.31: Heading Error during the Mountain Road Scenario

3.4.1 Low Speed

As a proof of the validity of our LKA, we here present the results obtained running the very same scenario with a target goal of **20 km/h** instead of 50 km/h.

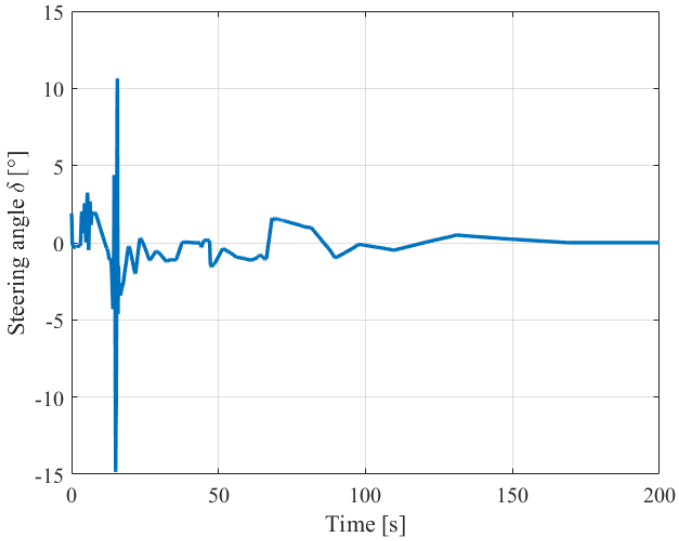

Figure 3.32: Steering Command during the Mountain Road Scenario

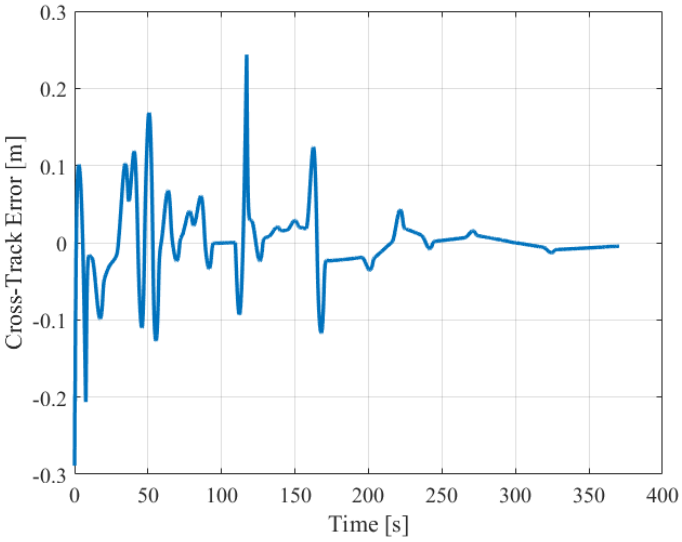

Figure 3.33: Cross-Track Error with a Target Speed $V_{des} = 20km/h$

3.5 [Mountain Road Circuit

Since this Scenario was a simple extension of the previous **Mountain Road** Scenario, we are giving below the graphs with the results of the Mountain Road Circuit Scenario without further comment.

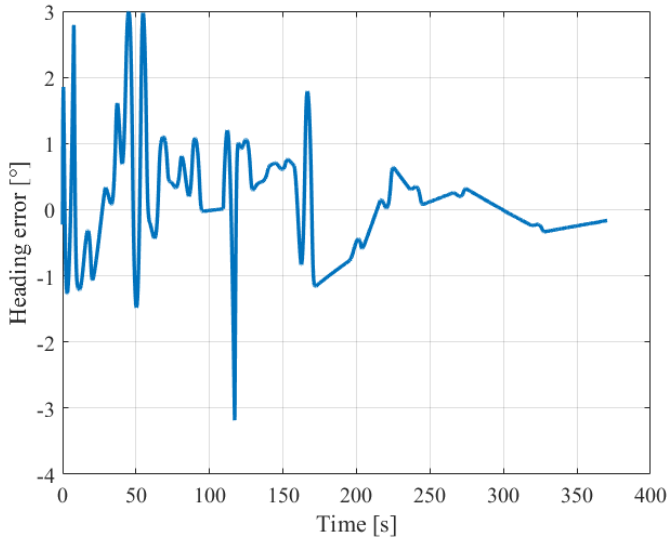

Figure 3.34: Heading Error with a Target Speed $V_{des} = 20km/h$

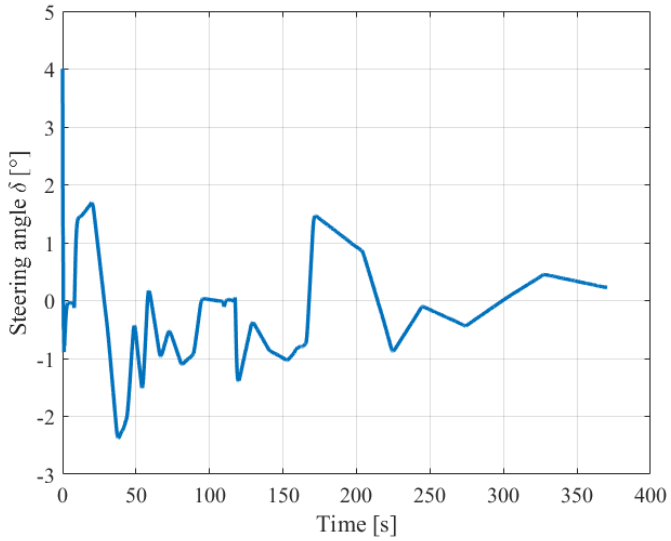

Figure 3.35: Steering Command with a Target Speed $V_{des} = 20km/h$

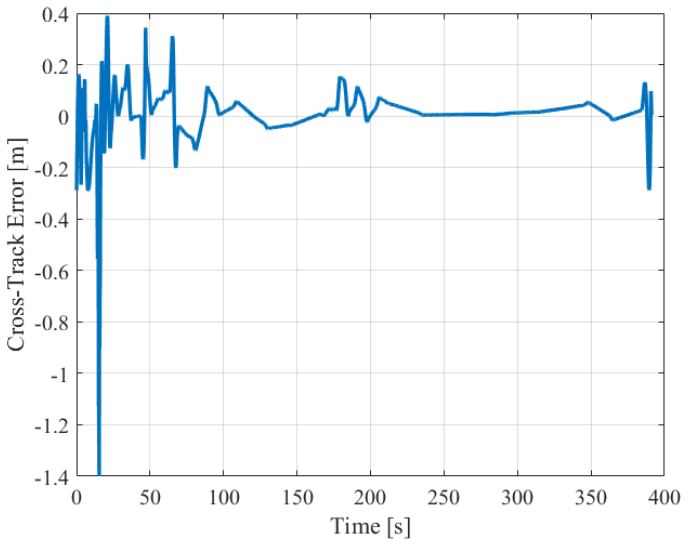

Figure 3.36: Cross-Track Error during the Mountain Road Circuit Scenario

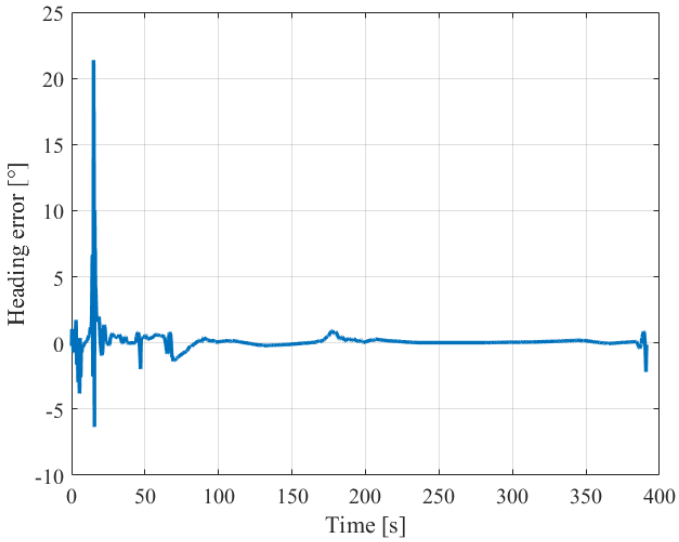

Figure 3.37: Heading Error during the Mountain Road Circuit Scenario

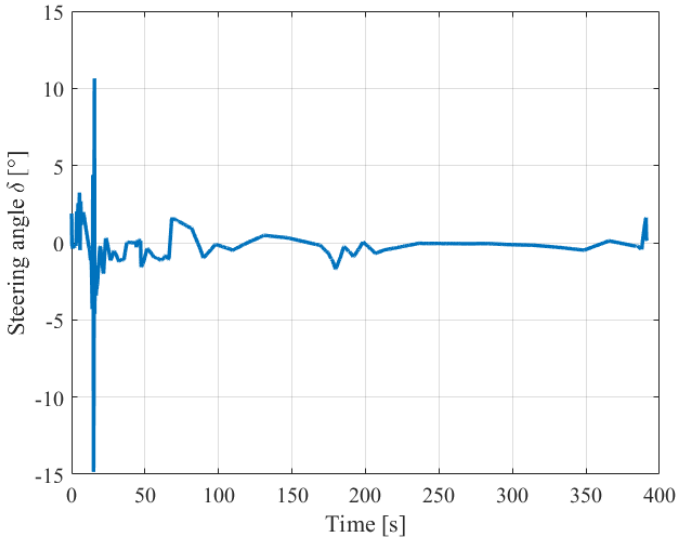

Figure 3.38: Steering Command during the Mountain Road Circuit Scenario

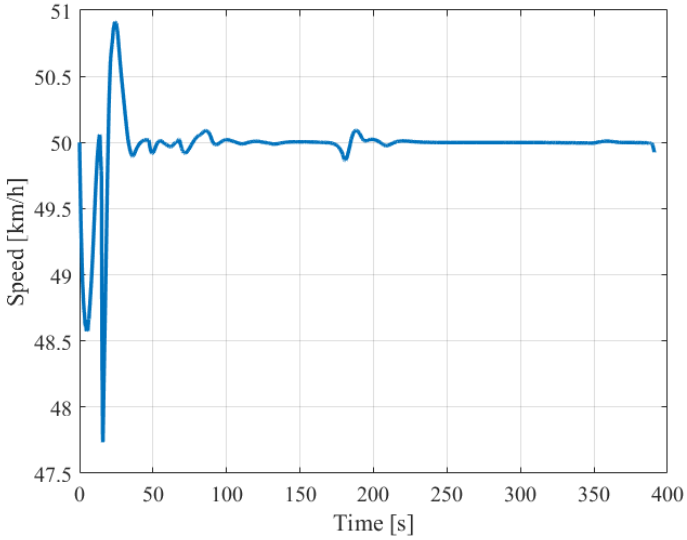

Figure 3.39: Speed of the EgoVehicle during the Mountain Road Circuit Scenario

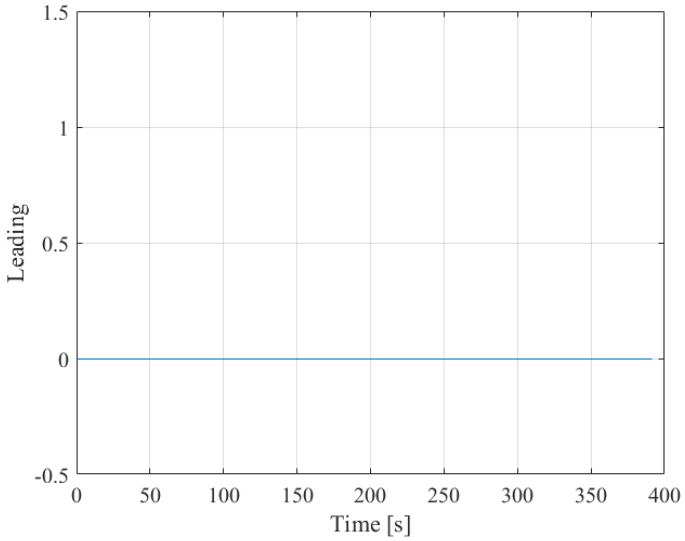

Figure 3.40: Leading Flag during the Mountain Road Circuit Scenario

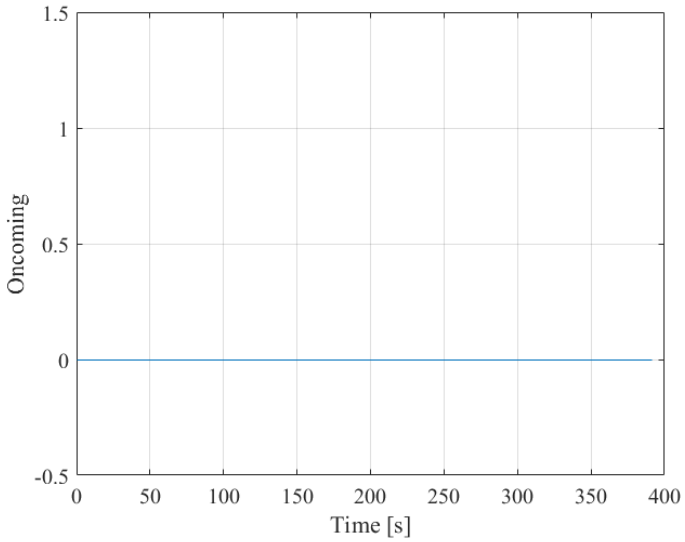

Figure 3.41: Oncoming Flag during the Mountain Road Circuit Scenario

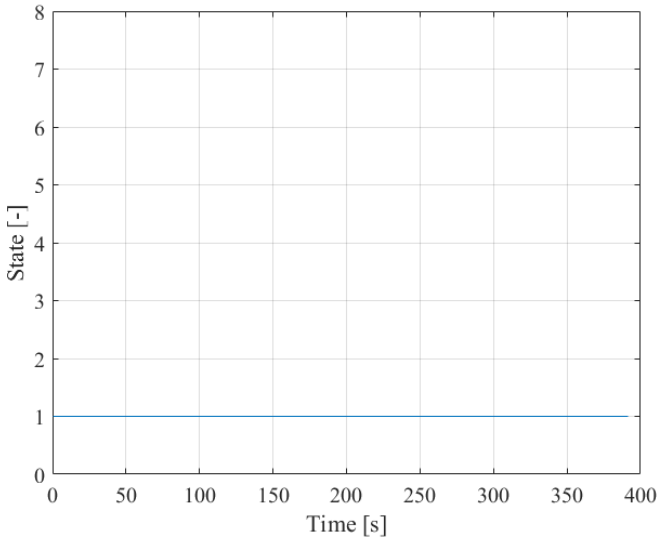

Figure 3.42: State during the Mountain Road Circuit Scenario

With these results, we consider the LKA system to be validated and, therefore, we move onto the validation of the Overtaking.

3.6 Overtake Scenario

Even if this Scenario is quite simple, in our opinion is probably the most important Scenario, as it is the one we used to test the Autonomous Overtaking Path Planning as well as the Decision Making State Machine: in fact, this was the first Scenario which we used for the autonomous Overtaking and all the following ones were built onto this to check that further complicating the Scenario would not damage the good working of our System.

The first thing which we want to comment is the trend of the State during the

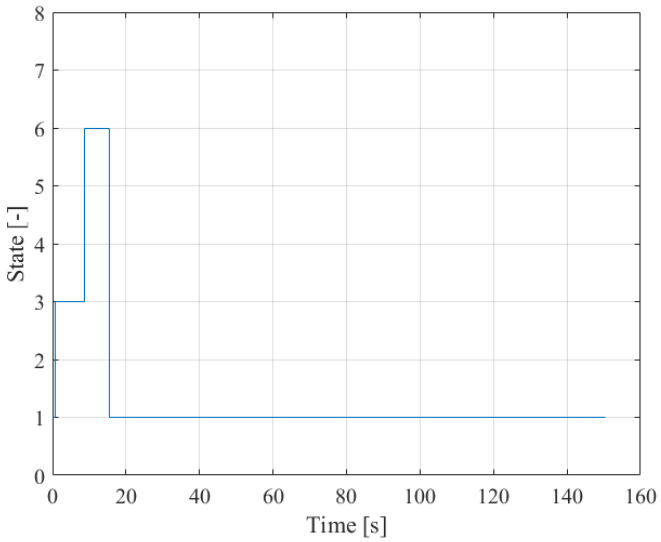

Figure 3.43: State during the Overtake Scenario

simulation: differently from what we saw in previous Scenarios - where the State was constantly in 1 (STAY) or eventually 2 (WAIT) - here we see that the State jumps into 3 (OVERTAKE), as we would expect and after some seconds moves to 6 (GO BACK) once the leading vehicle is overtaken. Eventually, the State settles into 1 (STAY) after the overtake is completed and the EgoVehicle is back in the OwnLane.

We can also see from Figure 3.43 that the State jumps to 3 from basically the first second of the simulation: this is because the leading vehicle is immediately in range of our sensors and this is confirmed by the trend of the Leading Flag in Figure 3.44 which is raised immediately.

Since the Scenario was supposed to be a simple scenario just to test the Autonomous Overtaking Path Planning and the Decision Making, we decided not to put any oncoming vehicle, so that we were not bothered by the OvtCounter (Section 2.7.1) and by sudden Abort maneuvers or Emergency Overtakes: this can be confirmed

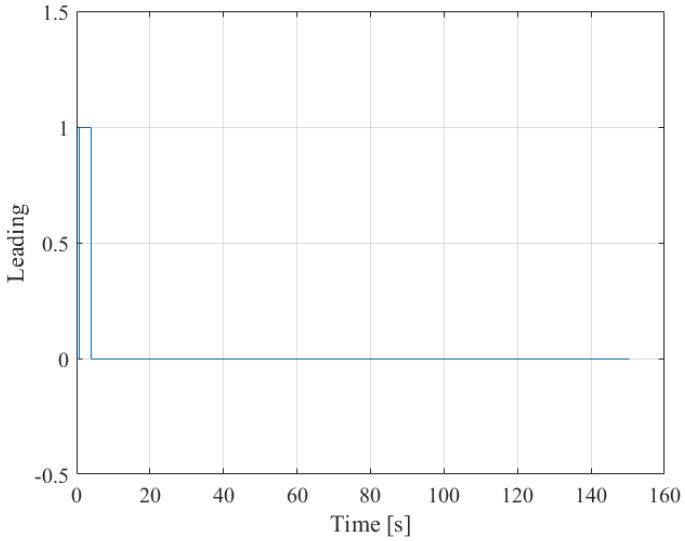

Figure 3.44: Leading Flag during the Overtake Scenario

by looking at Figure 3.45, which depicts an Oncoming Flag constantly equal to 0. Another important thing we could comment are the signals of the Radar, meaning

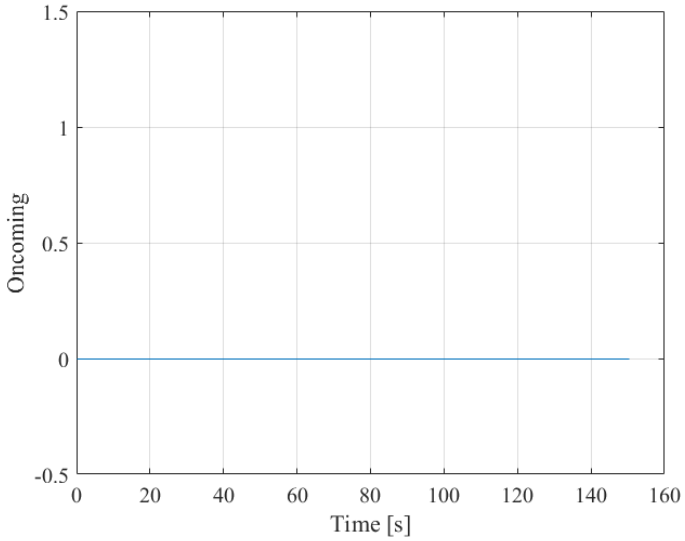

Figure 3.45: Oncoming Flag during the Overtake Scenario

the relative distance ΔX and ΔV relative speed, which are represented in Figure 3.46: we can see that the relative distance is steadily dropping while the relative speed is not changing a lot: this is not due to the leading vehicle inreasing its speed,

but simply to the fact that, since we are in State 3 (OVERTAKE) and not in State 4 (EMERGENCY OVT), the EgoVehicle is not forced to accelerate as much as possible, but tries to keep the set speed V_{des}, as we can also see from Figure 3.48

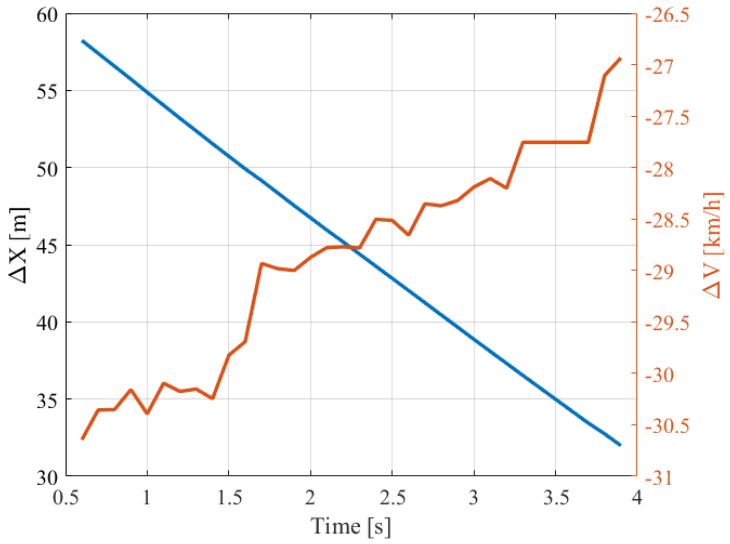

Figure 3.46: Radar signals during the Overtake Scenario

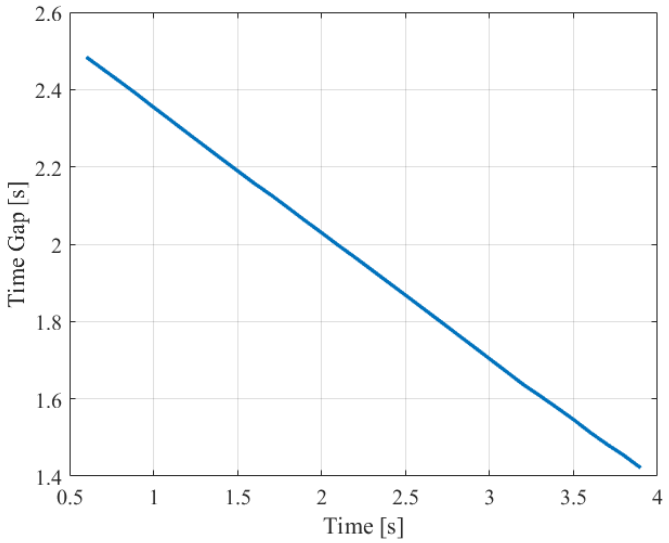

Figure 3.47: Time gap during the Overtake Scenario

where the initial drop is as always due to the starting acc command of 0: because

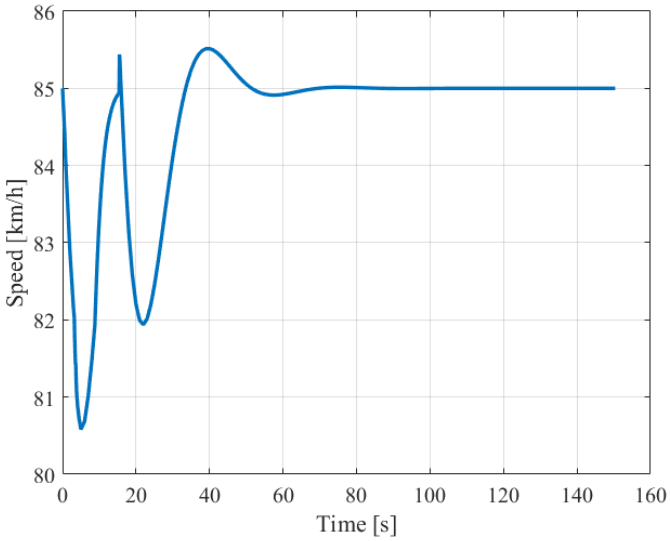

Figure 3.48: Speed of the EgoVehicle during the Overtake Scenario

of this, the ΔV is dropping through all of the time represented in Figure 3.46: this is because the Radar is tracking a leading vehicle only for the first 4-5 seconds, which is the portion of time in which the EgoVehicle speed drops; this limited time of the leading vehicle tracking explains why the Leading Flag duration is much shorter than the State 3 OVERTAKING, as we can see by comparing with the e_h which is an indication of our yaw angle ψ: the peak up to 7° of e_h is due to the car heavily steering towards the left, therefore leading the camera to see the Lane Boundaries going towards the right. The most important consequence of the vehicle steering, however is that the leading vehicle exits from the Field of View of the Central Radar and, therefore, the Leading Flag falls.
The heading error peak is, however, something which we cannot avoid, since the vehicle is not equipped with four wheel steering (which would allow to change lane without yawing) and we want to indeed change the lane to perform the Overtake. The good part is that the e_h rapidly falls to 0, meaning that the EgoVehicle quickly gets stabilized in the Oncoming Lane. Around the 10 s mark we see a negative spike in the e_h: this means that the Overtaking has been completed and we are now steering right to go back in the OwnLane. This is confirmed by the fact that at around the same 10 s mark, the State passes from 3 to 6 (GO BACK).
The Cross-Track Error displays the same trend as the Heading Error, even though by looking at the graphs of Figures 3.49 and 3.50 we could be misled into thinking the opposite. However, the sharp fall from 1.5 m to -1.3 is due to the fact that the EgoVehicle is now in the Oncoming Lane, even though just on the right boundary of said Lane; because of this, we move from being on the very left boundary of the

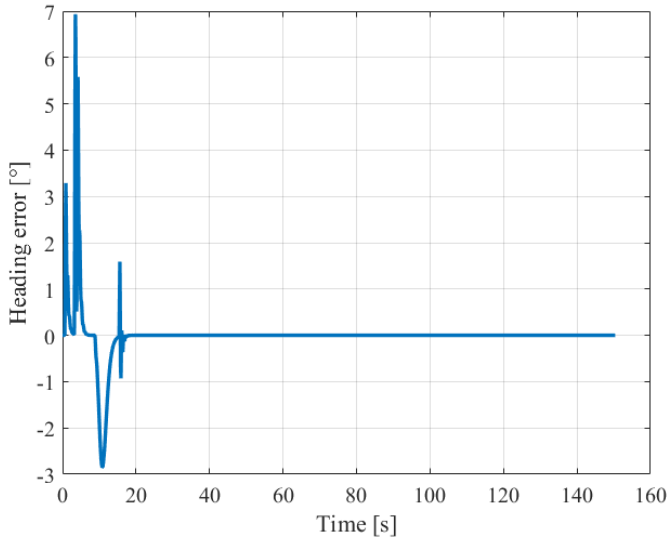

Figure 3.49: Heading Error during the Overtake Scenario

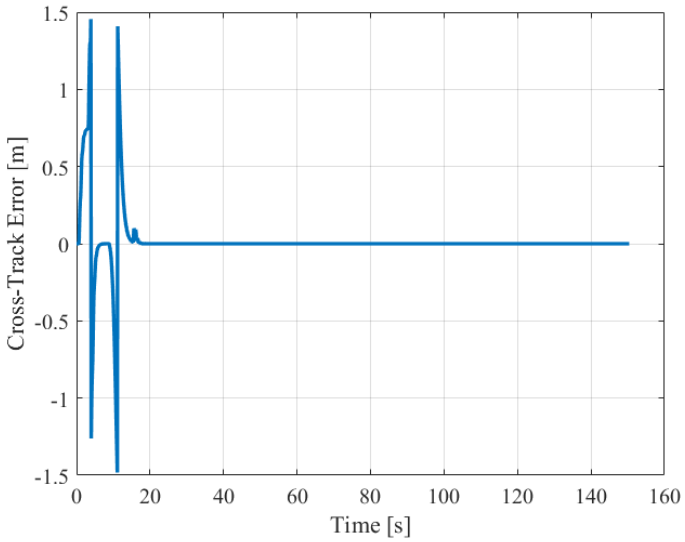

Figure 3.50: Cross-Track Error during the Overtake Scenario

Own Lane (causing $e_{ct} = +1.5m$) to being on theb right boundary of the Oncoming Lane (which should be causing $e_{ct} = -1.5m$, but is causing a local minimum around -1.3, due to the fact that the car is not parallel to the boundary lanes, leading to some error on the e_{ct} perception.
As a confirmation of this explanation, we see that the error on the Go Back

movement (around the 13 s mark), which is reversed, so first a negative error and then a positive one, is much closer to 1.5 m in both the peaks, since the Heading Angle is less than half in the second Lane Change. To conclude this Section, we

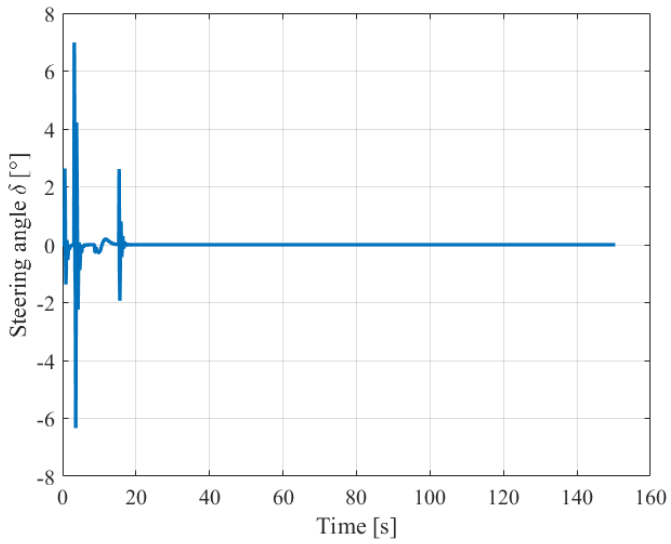

Figure 3.51: Steering Command during the Overtake Scenario

are going to present the trajectory followed by the EgoVehicle during this Scenario run: the Vehicle performs a sharp turn to the left in just 100 m, then stays in the Oncoming Lane and begins the Go Back maneuver just after 200 m in total; from Figure 3.53 we can see that the EgoVehicle is back in the Own Lane after just 260 m, traveling a total of just 170 m in the Oncoming Lane. Even though this is of course caused by the large difference in speed between us and the leading vehicle, but we can state that the first Autonomous Overtaking went well and so we moved to more complicated Scenarios.

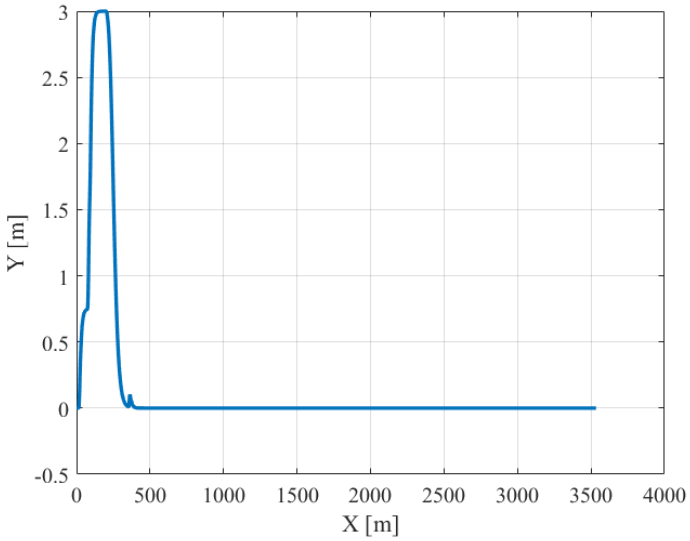

Figure 3.52: Trajectory followed during the Overtake Scenario

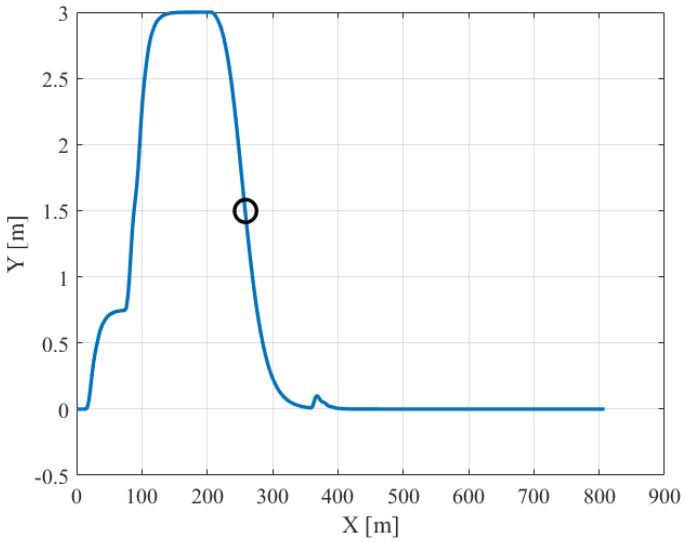

Figure 3.53: Zoom of the trajectory followed during the Overtake Scenario

3.7 Corner Overtake Scenario

In this Section we are going to discuss the results we obtained by the simulation of the Corner Overtake Scenario; as already stated in Section 2.4.7, we ran this Scenario twice, with the Lead Vehicle placed close to the EgoVehicle, therefore

entering immediately into the State 3 (OVERTAKE) and performing the overtake at the beginning of the turn and one where the Lead Vehicle is placed much further.

3.7.1 Early Overtake

The first thing we want to comment is the trend of the Leading Flag, which is raised immediately, as of Figure 3.54; this immediate raising of the Leading Flag

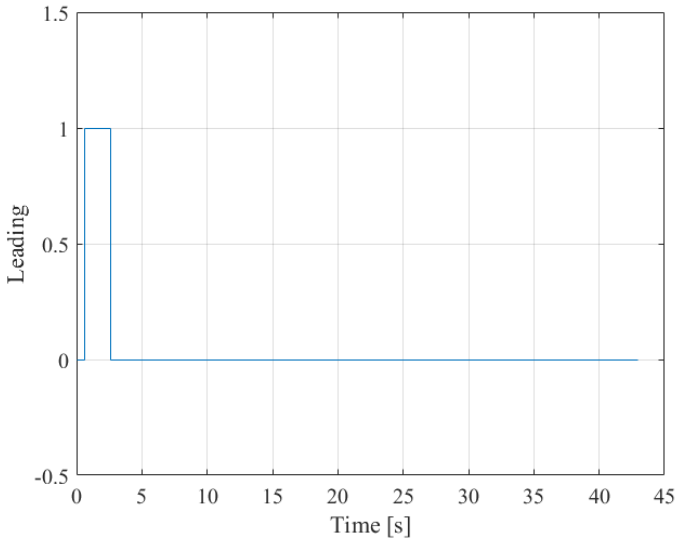

Figure 3.54: Trend of the Leading Flag during the Early variant of the Corner Overtake Scenario

is coherent with what can be seen in Figure 3.55, where the State immediately moves to 3 (OVERTAKE), since there is no oncoming vehicle (Figure 3.56 and the Centerline is dashed and not continuous. The fact that there is a second more or less before the State jumps to 3 (OVERTAKE) is due to the fact that it takes around a second for the GNN tracker to track the Radar detections and therefore for the Leading Flag to raise.
Following the discussion about the Leading Flag, we can present the signals of the Radar during the simulation, in Figure 3.57 the signals in Figure 3.57 are present only for about 2 seconds and this was already explained in Section 3.6: since the EgoVehicle is steering to the left in order to perform the Lane Change maneuver, the Lead Vehicle is not anymore in the Field of View of the Center Radar and therefore the radar is reporting only NaN values. Because of this, also the Time gap plot of Figure 3.59 is only present for 2 seconds.

The fact that the ΔV is dropping (i.e. its absolute value is decreasing, so the graph is pointing upwards) is, as always, due to the initial value of 0 for the acc

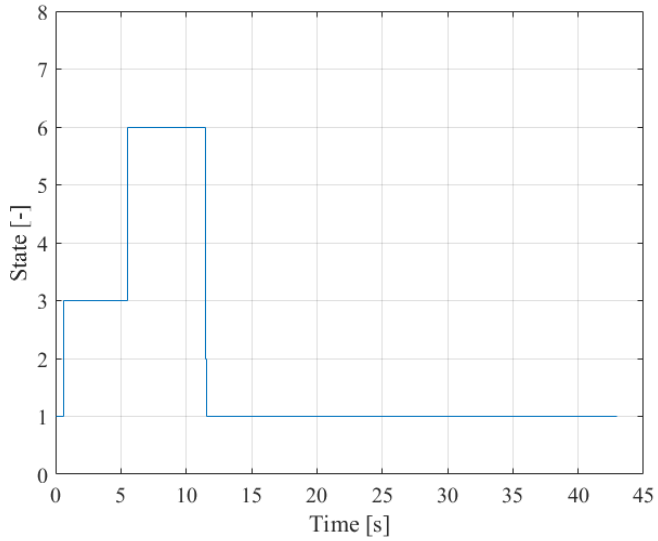

Figure 3.55: Trend of the State during the Early variant of the Corner Overtake Scenario

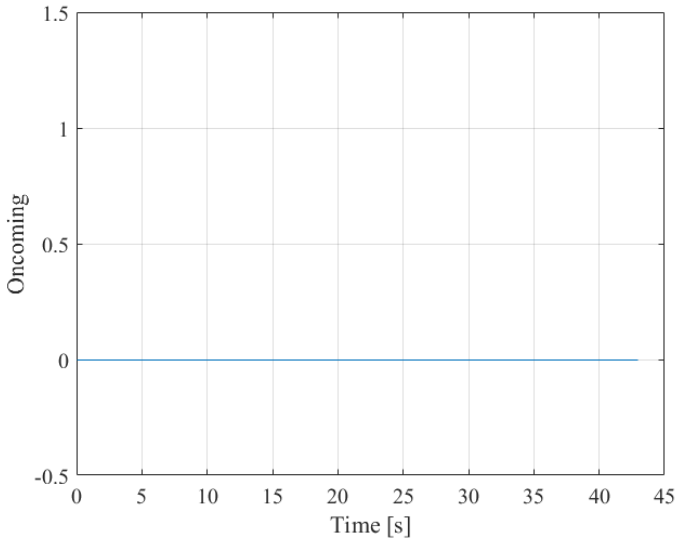

Figure 3.56: Trend of the Oncoming Flag during the Early variant of the Corner Overtake Scenario

command. Differently from what we stated for the first Scenarios, the errors are not directly linked anymore to the steering command, as the Sigmoid Curve is "correcting" the errors fed into the Stanley Controller, as in Figure 2.38: the errors

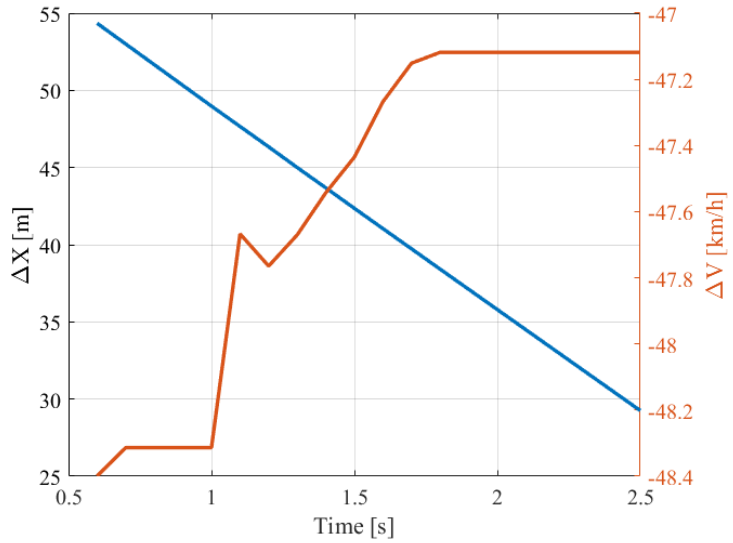

Figure 3.57: Radar signals during the Early variant of the Corner Overtake Scenario

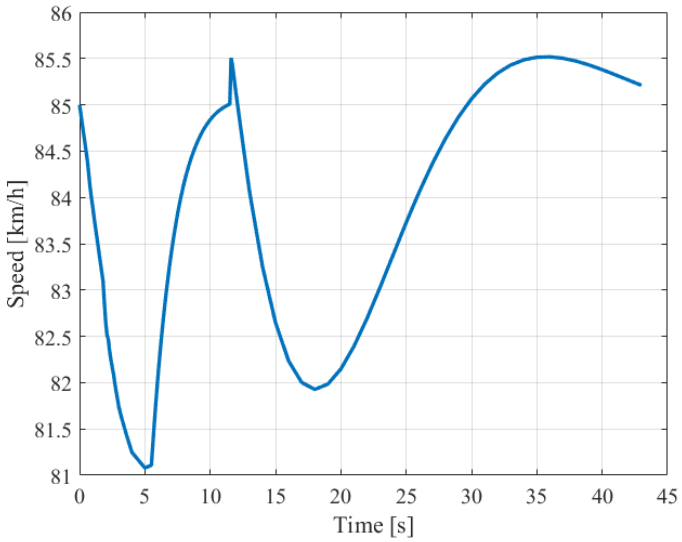

Figure 3.58: Speed of the EgoVehicle during the Early variant of the Corner Overtake Scenario

trend (Figures 3.60 and 3.61) are similar to those of Section 3.6.

As already said, the Cross-Track Error rises to 1.5 m before immediately dropping to -1.5 m in correspondence of the crossing of the Centerline; comparing the graphs

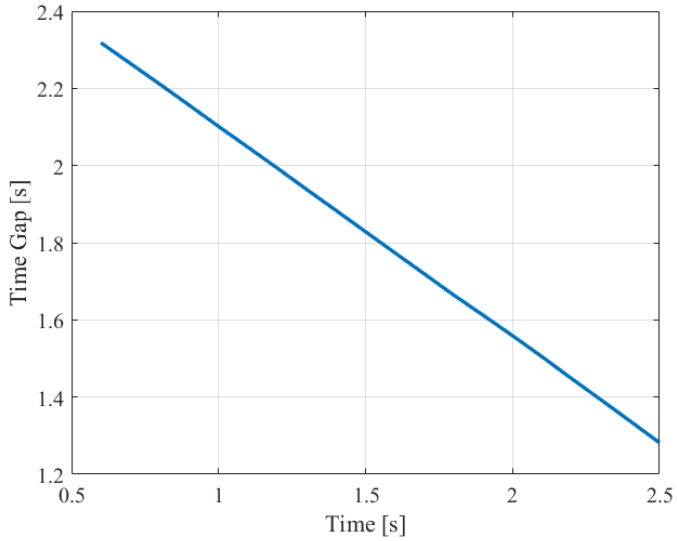

Figure 3.59: Time gap during the Early variant of the Corner Overtake Scenario

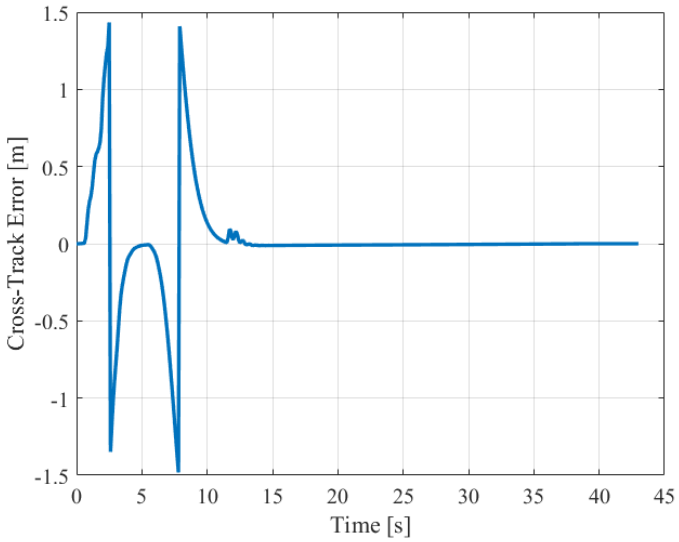

Figure 3.60: Cross-Track Error during the Early variant of the Corner Overtake Scenario

of Figure 3.50 and Figure 3.60 we can see that in the Corner Overtake Scenario the car does not even stabilize in the center of the Oncoming Lane, as can be seen by the knee around the 5 s mark of Figure 3.60, while in Figure 3.50 we see a stabilization of the vehicle, with nil e_{ct} and e_h: this, together with the reduced

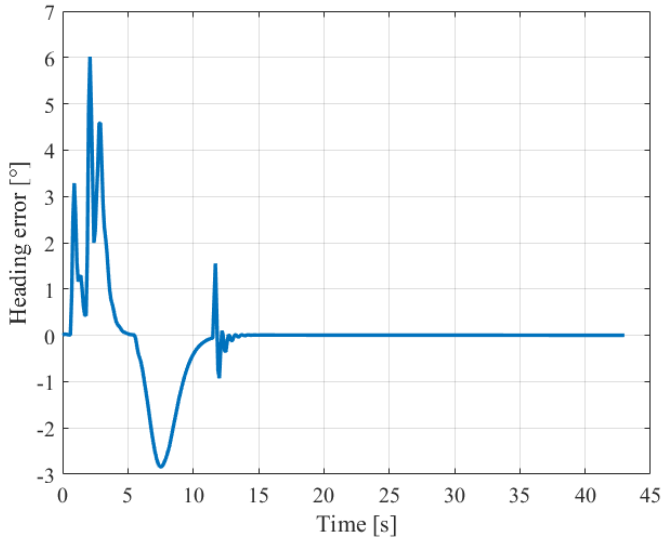

Figure 3.61: Heading Error during the Early variant of the Corner Overtake Scenario

duration of the State 3 (OVERTAKE) in the Corner Overtake Scenario compared to the Overtake Scenario is not due to a different working of the FSM, but just to the lower speed of the leading vehicle in the former (36 km/h compared to 54 km/h).
The steering command experienced in this Scenario is consistent with what we expect and what we already saw in previous Scenarios, namely Skidpad_Scenario and Overtake Scenario: a high peak around $+5°$ with a smaller ripple to stabilize the EgoVehicle in the Oncoming Lane, the not constant angle in the time window between 3 s and 13 s is due to the fact that the road is not straight, as said, and bends a bit to the right before the main turn to the left; the Go Back maneuver is less sharp and it is reflected by the much narrower δ steering command swing $[-2°2°]$ and consequently by the smaller e_h.
To sum up the Scenario, we present the plot of the trajectory followed by the EgoVehicle and its superposition with the FSM in order to verify that the Lane Change maneuvers are happening in correspondence with the States 3 (OVERTAKE) and 6 (GOBACK).

3.7.2 Late Overtake

We are going to put here the results of the second run of this Scenario, with the leading vehicle starting with a larger advantage, so that the EgoVehicle performs the Overtake in the long sweeping left corner instead of the beginning of it. We

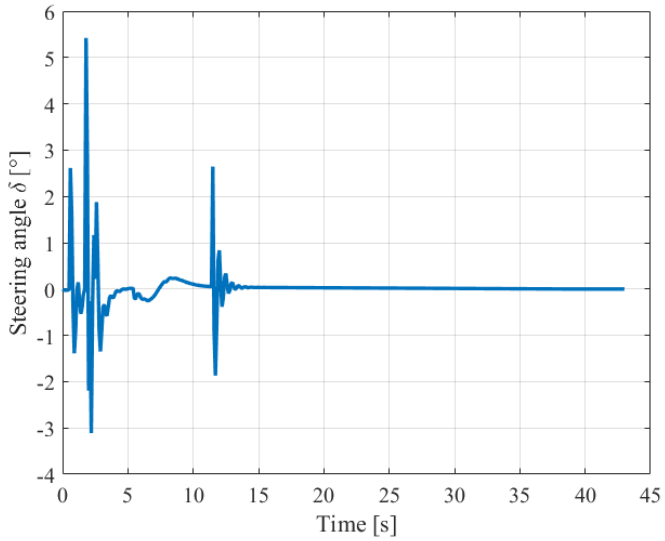

Figure 3.62: Steering Command during the Early variant of the Corner Overtake Scenario

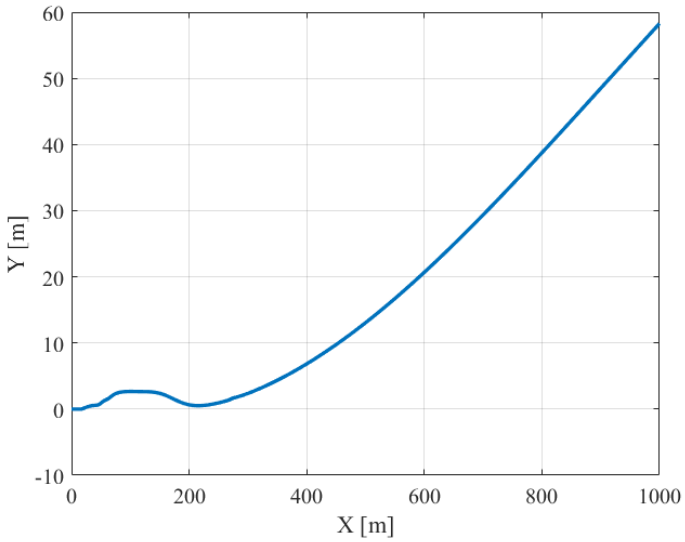

Figure 3.63: Trajectory followed by the EgoVehicle during the Early variant of the Corner Overtake Scenario

can see (Figure 3.65) that the Leading Flag is not raised immediately, of course, as the leading vehicle starts outside of the range of our sensing equipment. Once again, from Figure 3.66 we can see that there are no oncoming vehicles: we wanted

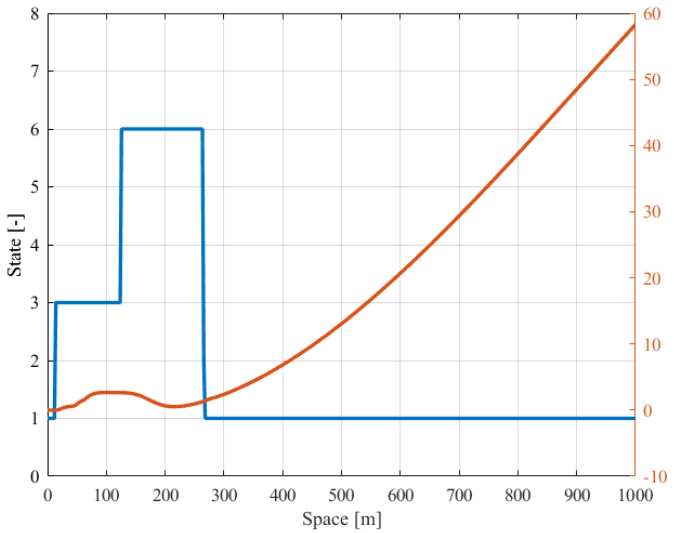

Figure 3.64: State and Trajectory during the Early variant of the Corner Overtake Scenario

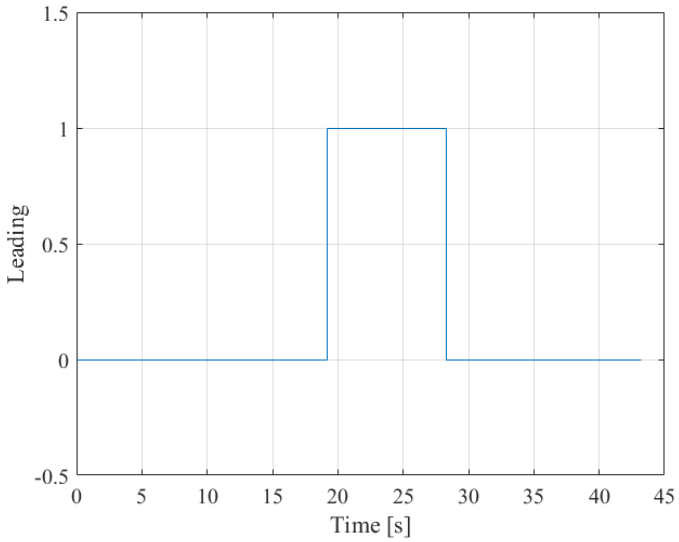

Figure 3.65: Trend of the Leading Flag during the Late variant of the Corner Overtake Scenario

to present this graph too, in order to prove that the data analysis can recognise a Vehicle as leading vehicle even if it is not perfectly aligned with us, provided it stays in the FoV of the Central Radar.

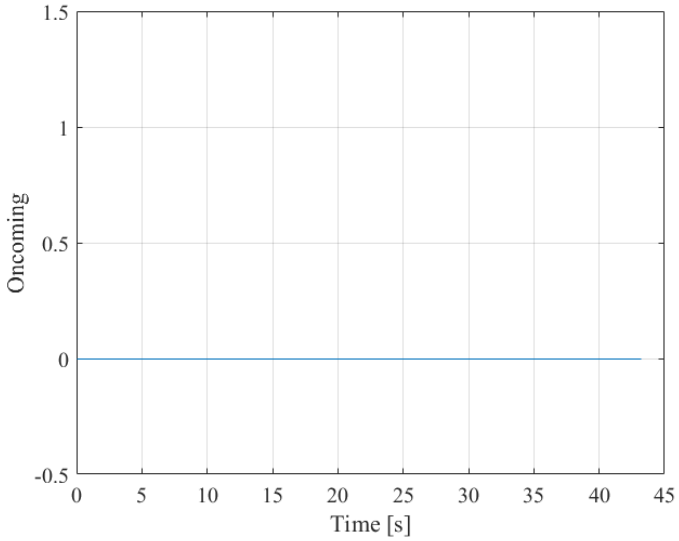

Figure 3.66: Trend of the Oncoming Flag during the Late variant of the Corner Overtake Scenario

The fact that there is no oncoming vehicle - and that the leading vehicle is much slower than us - leads to the State moving immediately from 1 to 3 as soon as the Leading Flag gets raised, as we can see comparing Figures 3.65 and 3.67. The fact

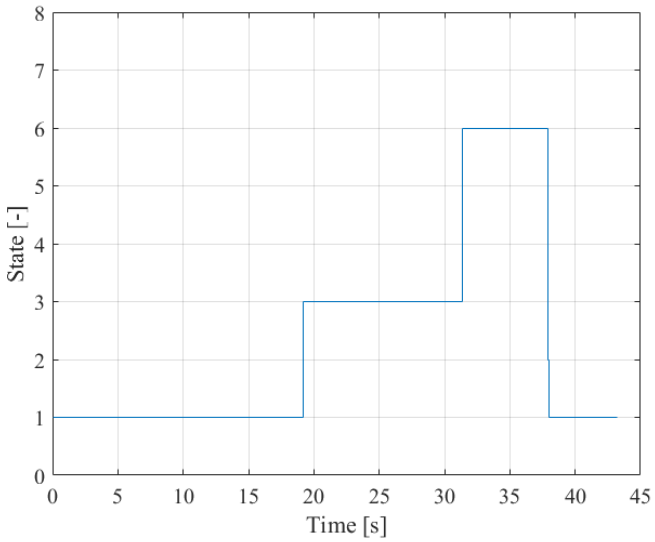

Figure 3.67: Trend of the State during the Late variant of the Corner Overtake Scenario

that we start with the LEading Vehicle outside of our sensing equipment means that such Lead Vehicle is viewed for a longer time, leading to a longer time range for the Radar signals, as is seen by comparing Figures 3.57 and 3.68. Anyway,

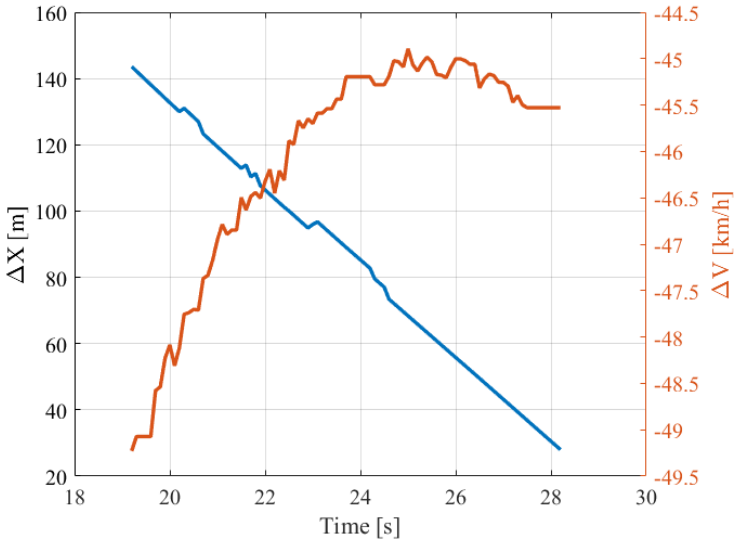

Figure 3.68: Radar signals during the Late variant of the Corner Overtake Scenario

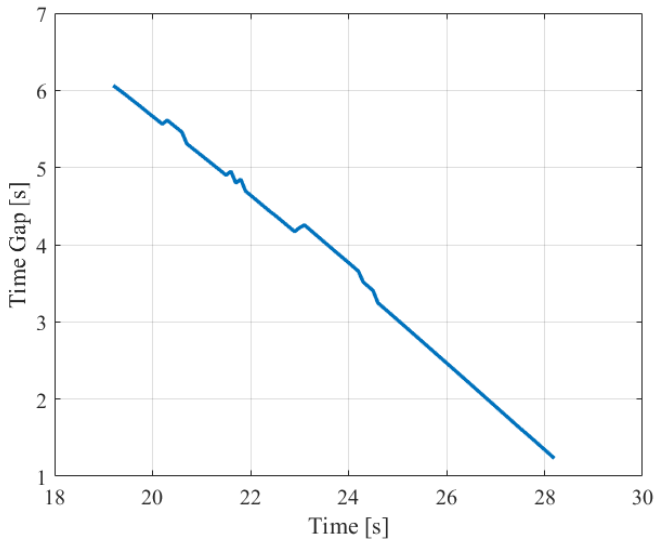

Figure 3.69: Time gap during the Late variant of the Corner Overtake Scenario

when the EgoVehicle is still around 25-30 m, it starts turning and this leads to

the signals of the Radar disappearing: this can be seen by a jump in the steering command around the 28 s mark in Figure 3.70, where the EgoVehicle starts the Lane Chenge maneuver. The other two jumps are in correspondance of the very beginning of the OVERTAKE state, when the leading vehicle appears in the sensing FoV and this forces the EgoVehicle to move towards the center of the road (20 s) and to the end of the OVERTAKE State, when the EgoVehicle moves into State 6 (GO BACK) around the 37 s mark. Anyway, we can see that the steering command

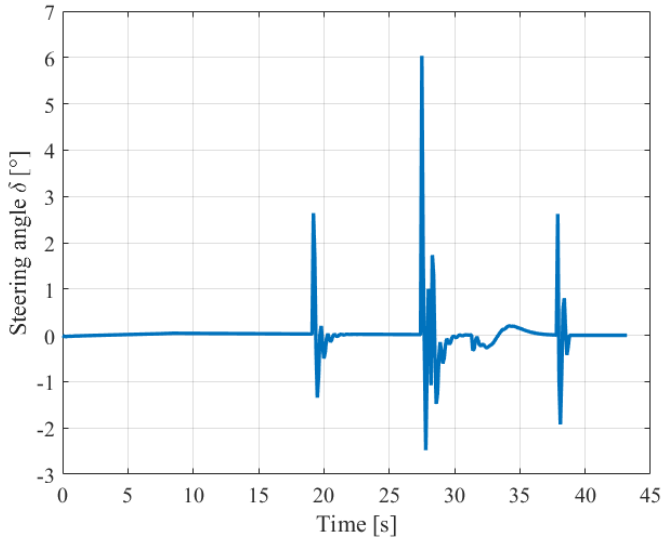

Figure 3.70: Steering Command during the Late variant of the Corner Overtake Scenario

is in general acceptably low, confirming that we avoided a too sharp maneuver. This is further confirmed by Figures 3.71 and 3.72, which do not display a significant noise in the errors. The large plateau in the Cross-Track Error of Figure 3.71 is due to the fact that, since differently from the previous Scenarios here we do not start already in Overtaking Range, we move towards the center of the road and stay there until we reach the Overtake Range and it takes longer here, so we can detect it here, compared to Figure 3.50. In order to ensure the comfort of the passengers, we aim to not only keep the car stable laterally, but also to reduce as much as possible its longitudinal jerk. This requirement is verified from the results found in Figure 3.73, where we see that - apart from the initial drop due, as always to the starting acc command of 0 - the speed is pretty much constant around the desired value of $85 km/h$, as of Figure 3.73. To sum up this run, we are going to display the trajectory followed by the EgoVehicle (Figure 3.74) as well as a plot displaying an overlay of the State with the trajrctory (Figure 3.75).

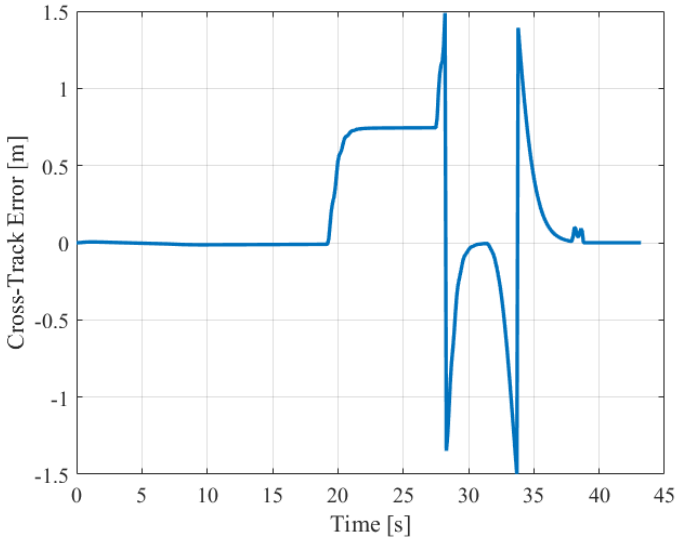

Figure 3.71: Cross-Track Error during the Late variant of the Corner Overtake Scenario

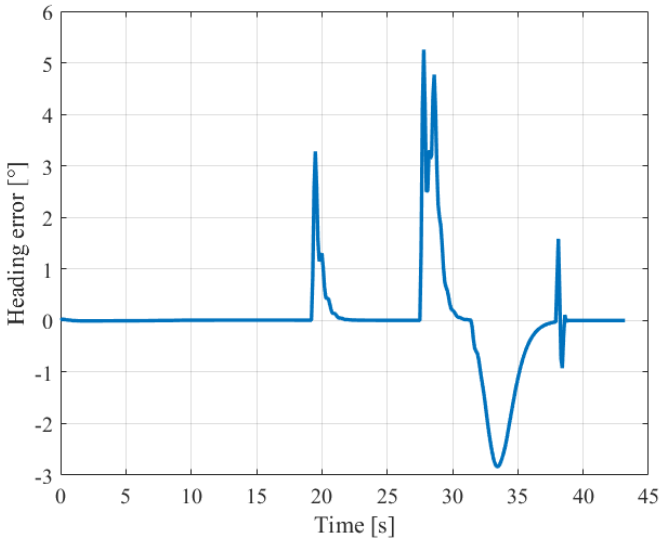

Figure 3.72: Heading Error during the Late variant of the Corner Overtake Scenario

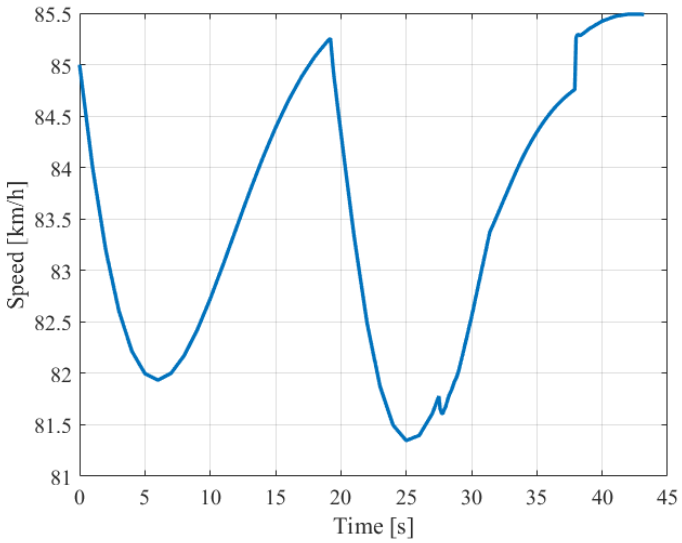

Figure 3.73: Speed of the EgoVehicle during the Late variant of the Corner Overtake Scenario

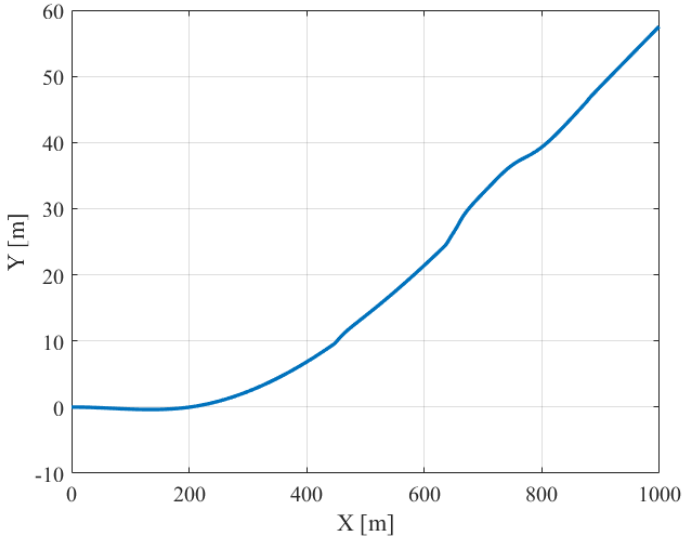

Figure 3.74: Trajectory followed by the EgoVehicle during the Late variant of the Corner Overtake Scenario

3.8 Double Overtake Scenario

Here are the results of the Double Overtake Scenario simulation; the first thing we notice, which we of course were expecting, is that the Radar signals (Figure

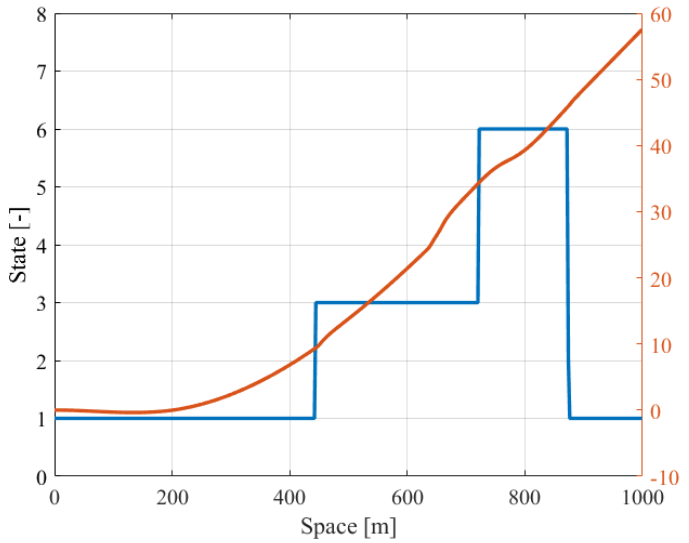

Figure 3.75: State and Trajectory during the Late variant of the Corner Overtake Scenario

3.76) and the Time gap (Figure 3.77 graphs have two separated moments with data each): this is perfectly natural and expected, since the EgoVehicle finds itself twice following a Lead Vehicle.

This is coherent with what we see in the State plot, where in fact we see repeated

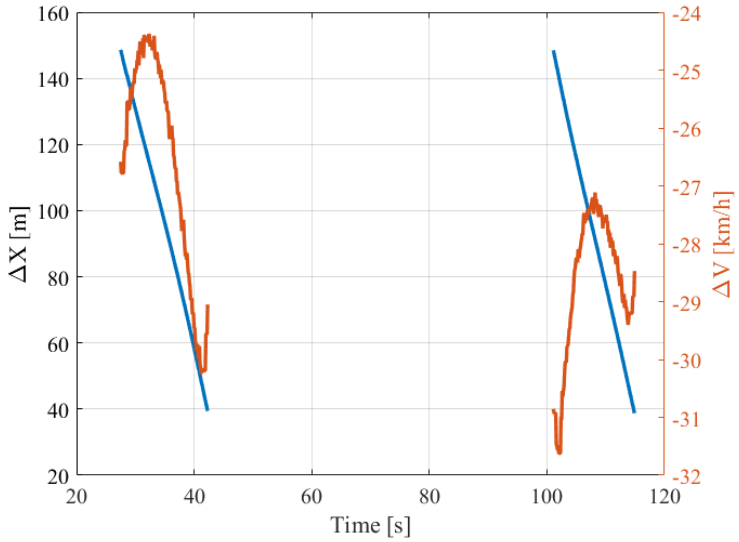

Figure 3.76: Radar signals during the Double Overtake Scenario

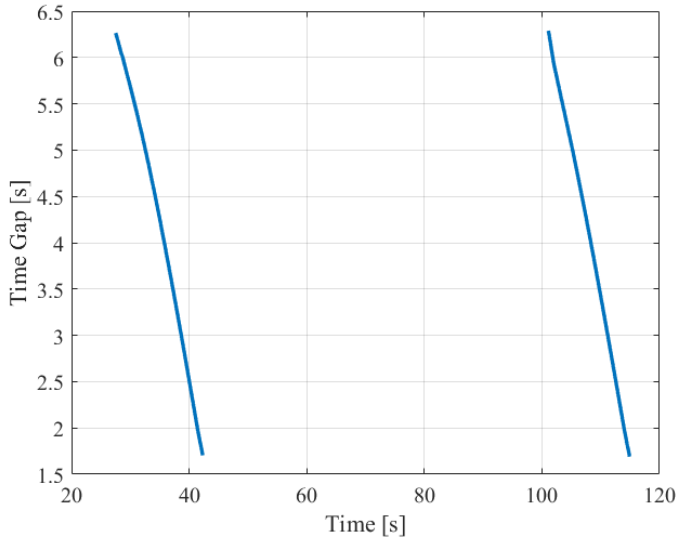

Figure 3.77: Time gap during the Double Overtake Scenario

twice the profile corresponding to the Overtake maneuver (compare Figure 3.78 and Figure 3.43). The double Overtake maneuver is also confirmed by what we

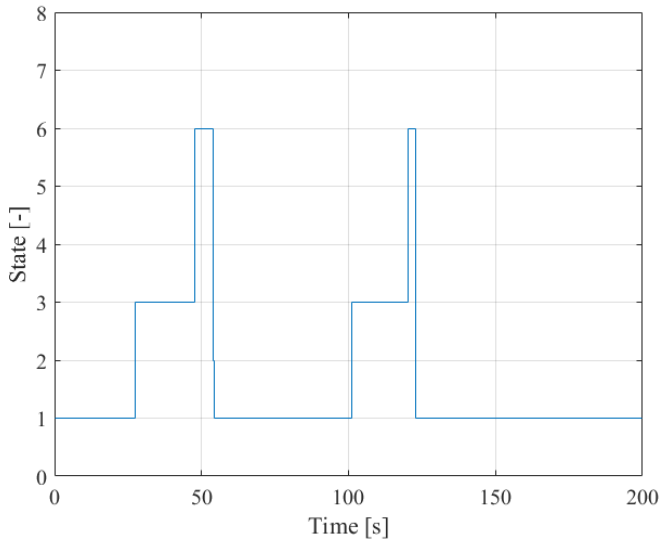

Figure 3.78: Trend of the State during the simulation of the Double Overtake Scenario

can see in the Leading Flag graph (Figure 3.79) where the Flag is raised twice, in correspondence of the . The fact that the EgoVehicle has to perform two

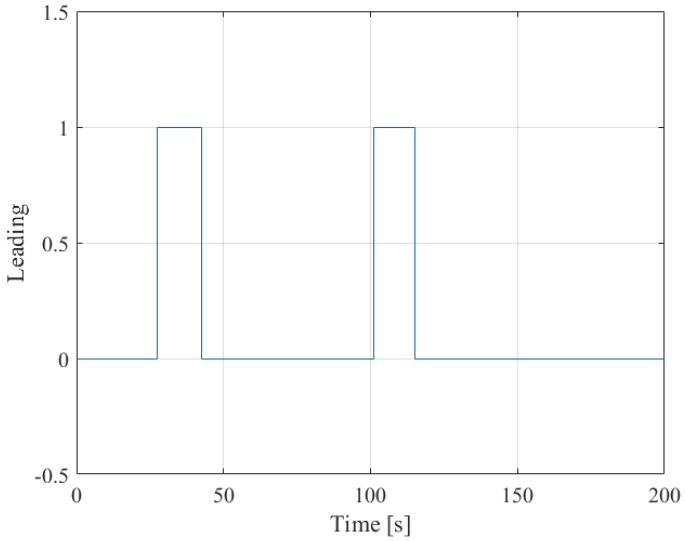

Figure 3.79: Trend of the Leading Flag during the Double Overtake Scenario

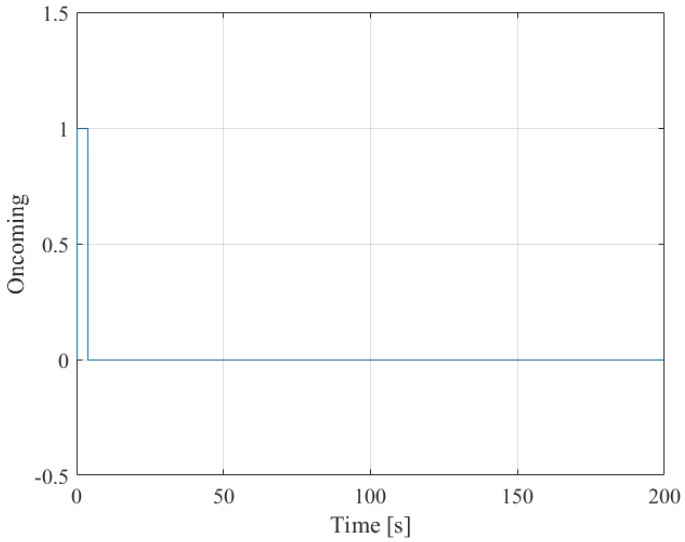

Figure 3.80: Trend of the Oncoming Flag during the Double Overtake Scenario

separate Overtake maneuvers does not hinder the stability of the EgoVehicle or its trajectory, as can be seen by checking Figures 3.81 and 3.82 which display Error trends similar to the ones seen in the Single Overtakes (Figures 3.50 and 3.49.

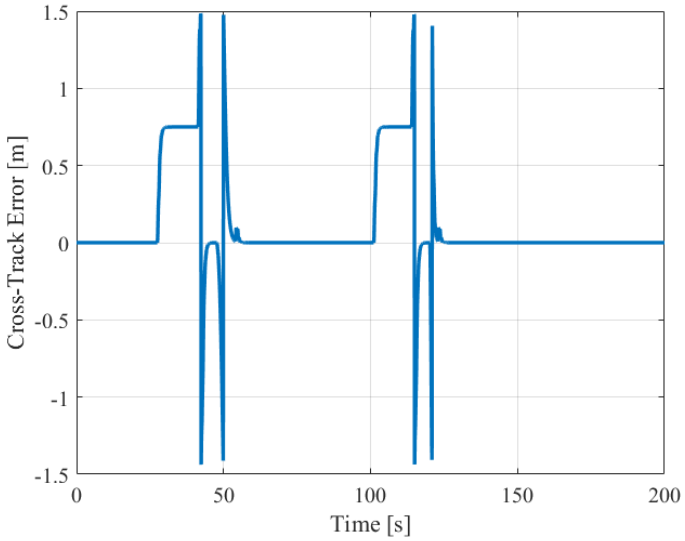

Figure 3.81: Trend of the Cross-Track Error during the Double Overtake Scenario

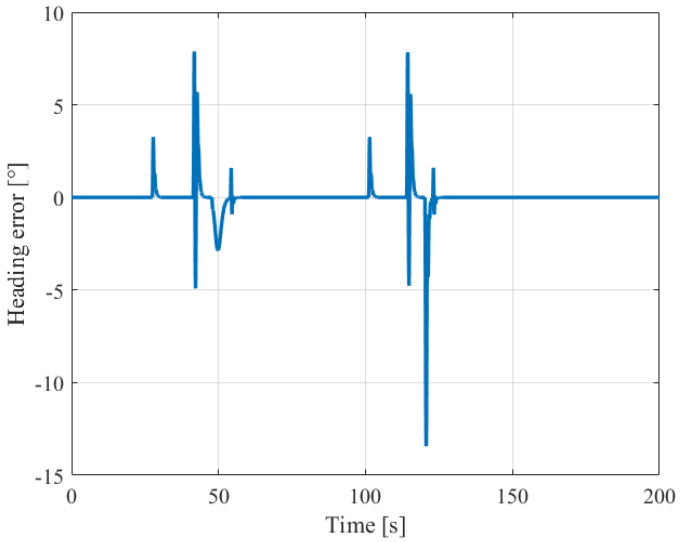

Figure 3.82: Trend of the Heading Error during the Double Overtake Scenario

To sum up the results of this Scenario, we want to present the trend of the speed of the EgoVehicle, which drops during the steering maneuver, and the trajectory followed by the EgoVehicle to check that the Overtakes are performed according to what we expect, as well as the Steering Angle trend.

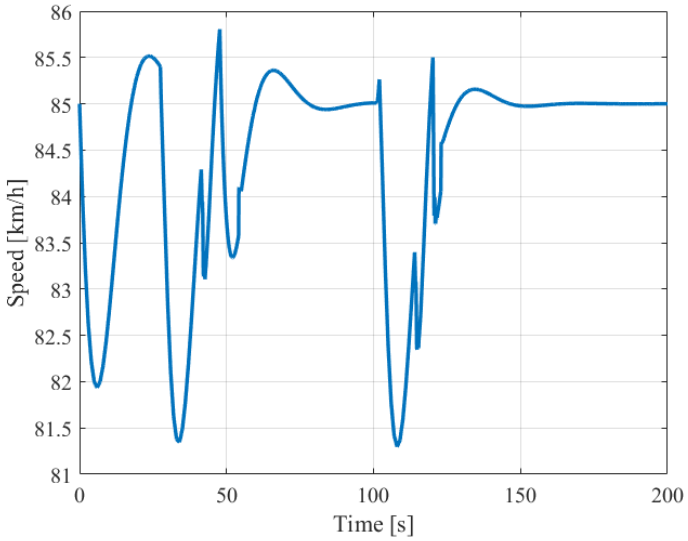

Figure 3.83: Speed of the EgoVehicle during the Double Overtake Scenario

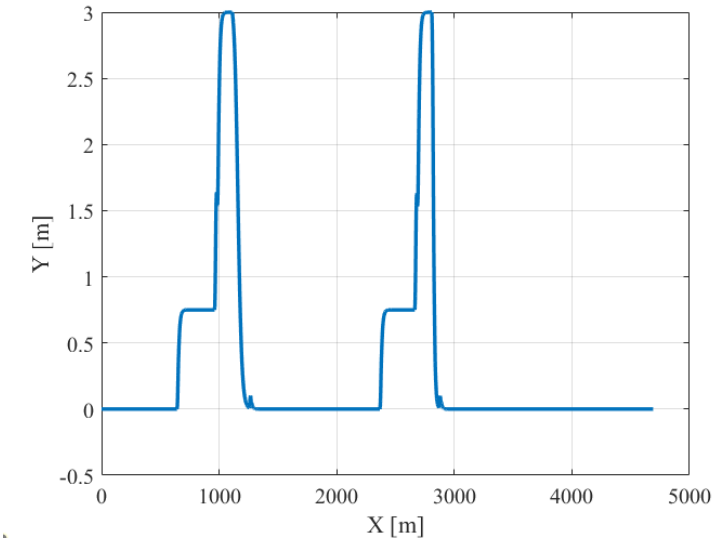

Figure 3.84: Trajectory of the EgoVehicle during the Double Overtake Scenario

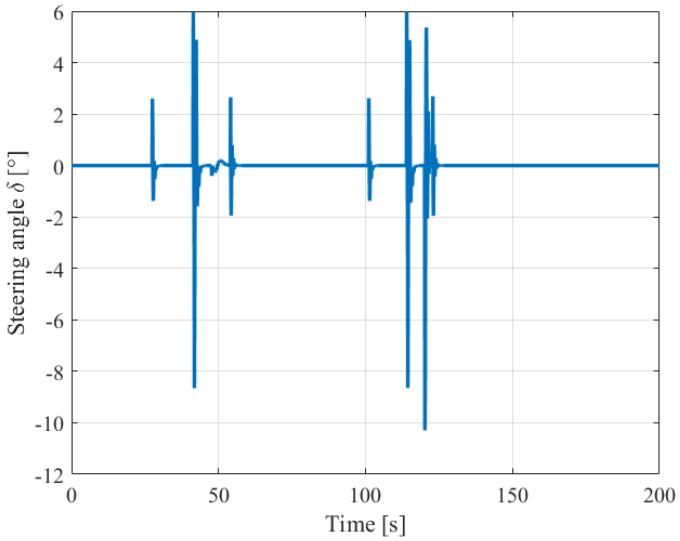

Figure 3.85: Steering Command during the Double Overtake Scenario

Finally, we highlight the trajectory as a function of the state in which the FSM is, to confirm the validity of our results.

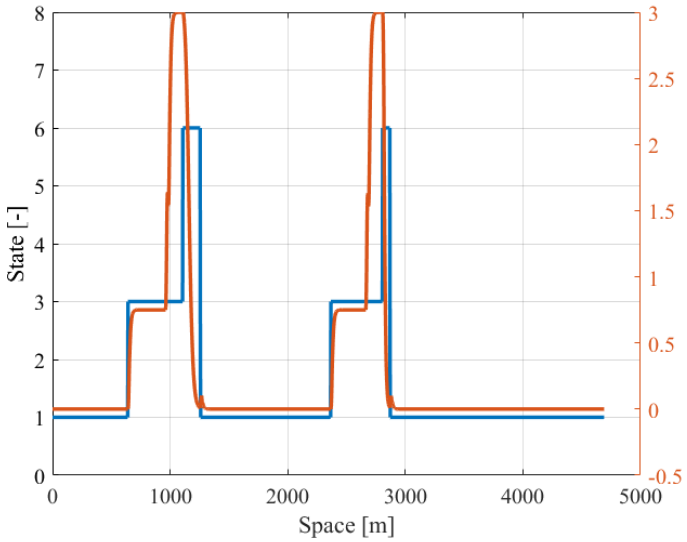

Figure 3.86: State and Trajectory of the EgoVehicle during the Double Overtake Scenario

3.9 Multiple Overtake Scenario

As we have already stated at the beginning of Section 3.8, we have a block of two periods in which we have signals from the radar. This might look as an error, as we have 3 cars to overtake, but there is a precise reason for this: first of all, we can see that the Central Radar is outputting results only in case we are in the Own Lane (OwnLane == 1, as per the coordinates of Figure 2.29), therefore the third leading vehicle is not reported by the Central Radar as we are still in the Overtake Lane. The reason which led us to this technique is the function $y = f(\Delta X)$: if we allowed the Center Radar to report the coordinates of the Lead Vehicle, as soon as the second Lead Vehicle to be overtaken comes into the Center Radar range, the Sigmoid computation would force the EgoVehicle to heavily steer towards the right, thus crashing into the first vehicle we are overtaking.

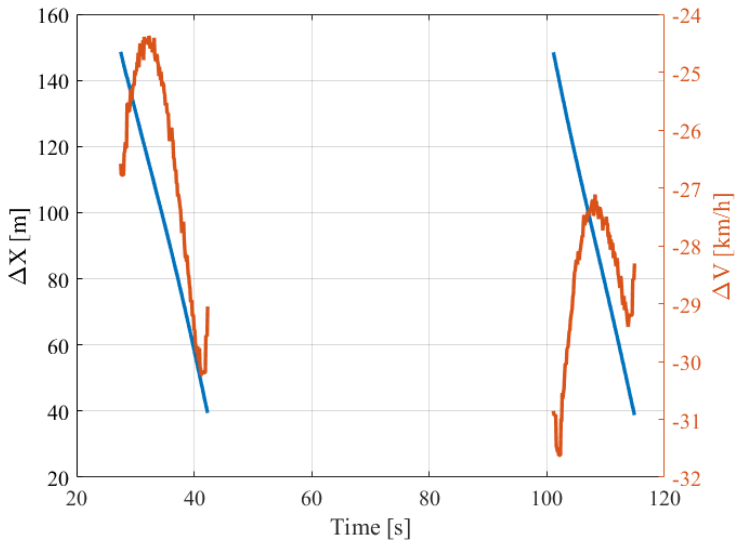

Figure 3.87: Radar signals during the Multiple Overtake Scenario

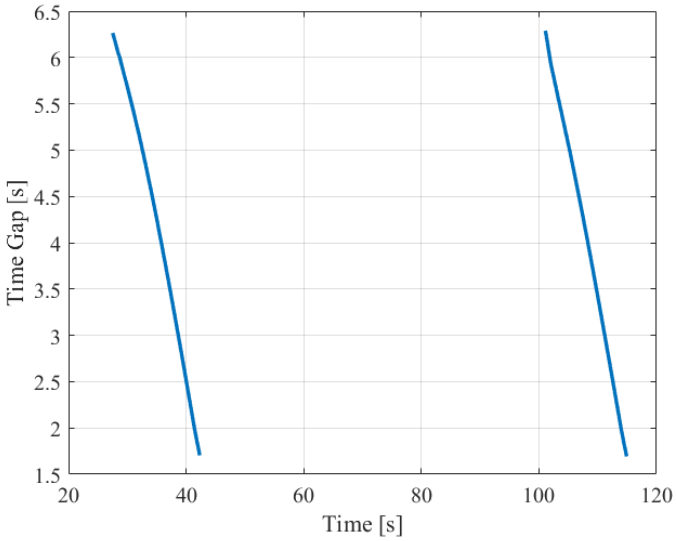

Figure 3.88: Time gap during the Multiple Overtake Scenario

To solve this problem, we fed the output of the central Radar GNN not only to the Central Radar data analysis itslef, but also to the "Overtaken" Matlab function block, together with the Right BlindSpot Radar: in this way, as we can see from the Algorithm of Section 2.7.6, the check on the Overtaken flag is performed on the presence of any Vehicle in the range of the Central Radar as well as the Right BlindSpot Radar; only if no Vehicle in the range of these two Radars has a relative X higher than $-3m$, the Overtaken Flag is raised and the egoVehicle can move into State 6 (GO BACK).

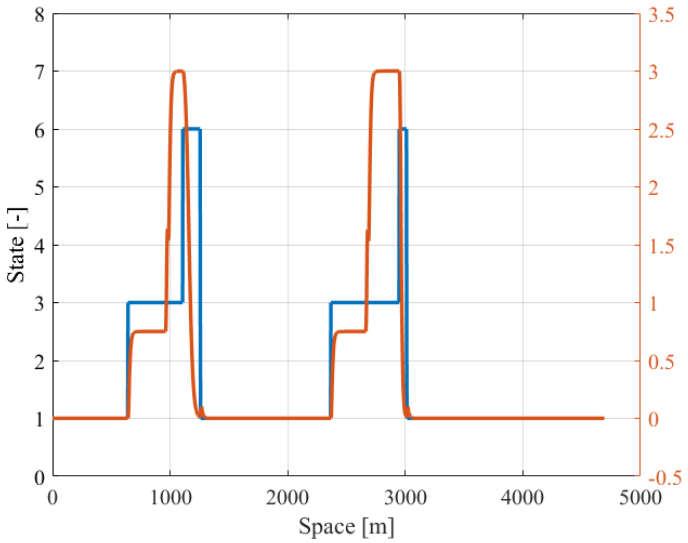

Figure 3.89: State and Trajectory of the EgoVehicle during the Multiple Overtake Scenario

A further confirmation of this is provided by Figure 3.89, where we see an overlay of the Vehicle trajectory onto the State trend: the time duration of State 3 (OVERTAKE) in the second Overtake maneuver, where we overtake the Second and the Third leading vehicles, is considerably longer than the one of the first Overtake maneuver, where there is a single Vehicle to Overtake and the same can be said of the length in meters, considering that the Speed of the EgoVehicle can be considered reasonably constant throughout the two maneuvers, as displayed by Figure 3.90.

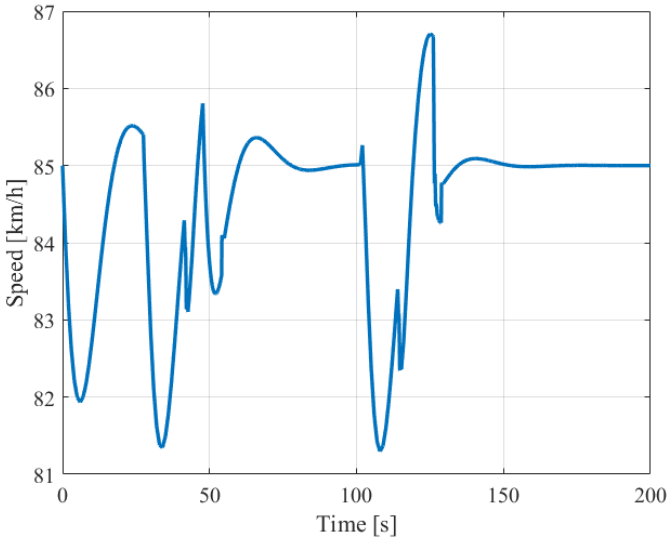

Figure 3.90: Speed of the EgoVehicle during the Multiple Overtake Scenario

The fact that the Radar Signals are not output by the Central Radar PostProcess in case we are in the Overtake Lane is also confirmed by the Oncoming Flag trend, displayed in Figure 3.91, where we can see that the Leading Flag stays up only while the EgoVehicle is in the OwnLane, close to the Lane marker, as confirmed by a comparison with Figure 3.89.

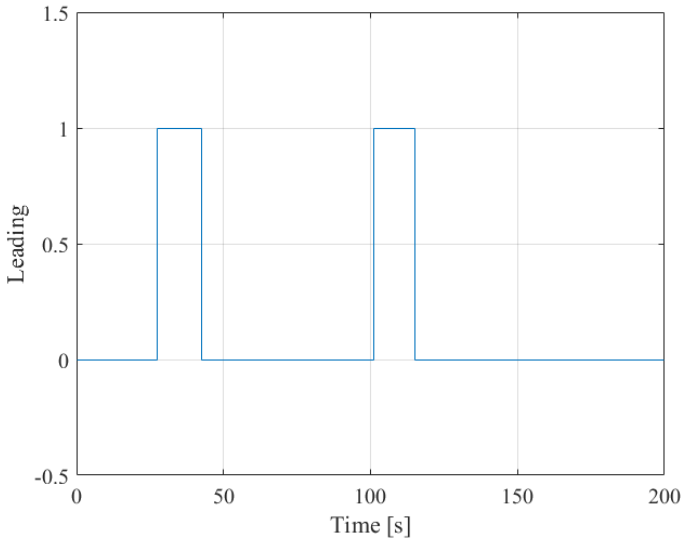

Figure 3.91: Trend of the Leading Flag during the Multiple Overtake Scenario

By the way, while the duration of the **State 3 (OVERTAKE)** is higher in the second maneuver, the case is reverted for the Leading Flag: this is caused by the fact that the second Lead Vehicle is notably slower than the first one on the approach phase, as can be seen by the value of ΔV in Figure 3.87, which also leads to a time gap signal dropping faster (Figure 3.88, hence the EgoVehicle catching up faster to the Lead Vehicle.
Meanwhile, in this Scenario too, the Oncoming Flag was fine, with only the real oncoming vehicle being recognized as such: this is the product of adjustments to the Central Radar data analysis (Section 2.7.5) in order to avoid misrecognitions.

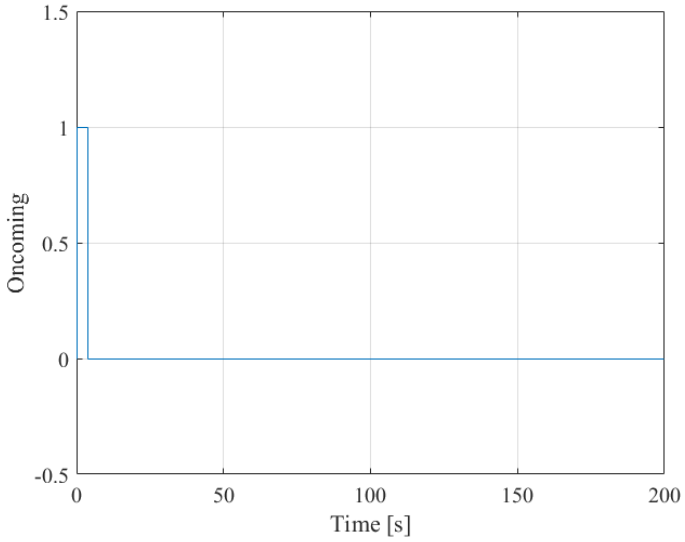

Figure 3.92: Trend of the Oncoming Flag during the Multiple Overtake Scenario

To further demonstrate that the Overtaken Flag works as intended, we are going to comment Figure 3.93: first of all, we see that after overtaking the Parked Truck on the right at the very beginning of the Scenario, the Overtaken Flag is raised (as should be, since we have indeed overtaken it!), but this does not lead to a random jump into State 6 (GO BACK), confirming that the Next State function for State 1 was properly designed, avoiding to drive off the road if we overtake a random vehicle parked on the side of the road.

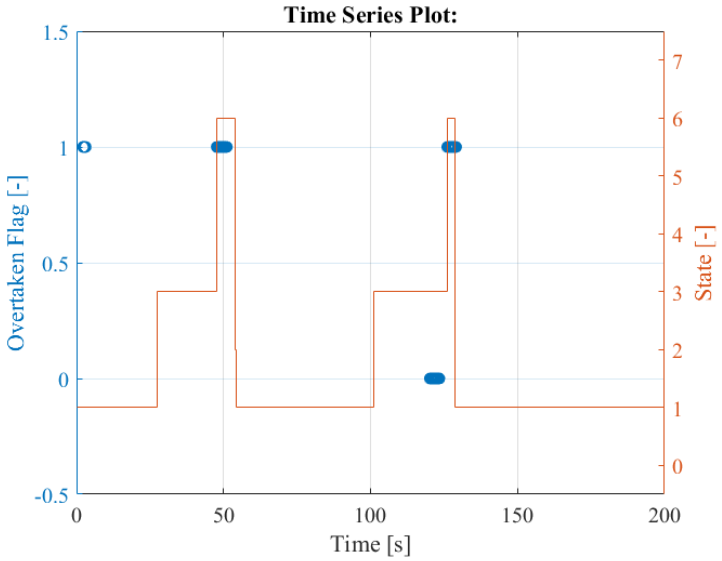

Figure 3.93: Comparison between the Overtaken Flag and the State during the Multiple Overtake Scenario

The first maneuver ends with the EgoVehicle jumping into State 6 (GO BACK), as expected, while the situation is different in the second Overtake maneuver: in this we can distinguish two different phases according to the Overtaken Flag:

- A first phase (120.2 s - 123.4 s), where the Overtaken Flag is still 0 and therefore the State remains at 3 (OVERTAKE).

- A second phase (126.1 s - 129.1 s), where the Overtaken Flag is now 1 and the State jumps immediately to 6 (GO BACK)

The first phase has the Overtaken Flag set to 0, since as said, the leading vehicle prevents this Flag to raise, while in the second phase we have overtaken the second of these two Vehicles (the Third leading vehicle from the tables of Section 2.4.9) and therefore we have no Lead Vehicle anymore and the Overtaken Flag can be raised; it is to be noted that the gap between these two as well as the gap between the two maneuver is an effect of the data analysis Algorithm, which is intended to return NaN as an Overtaken Flag in case we have no tracked Vehicle in any of the Radars.

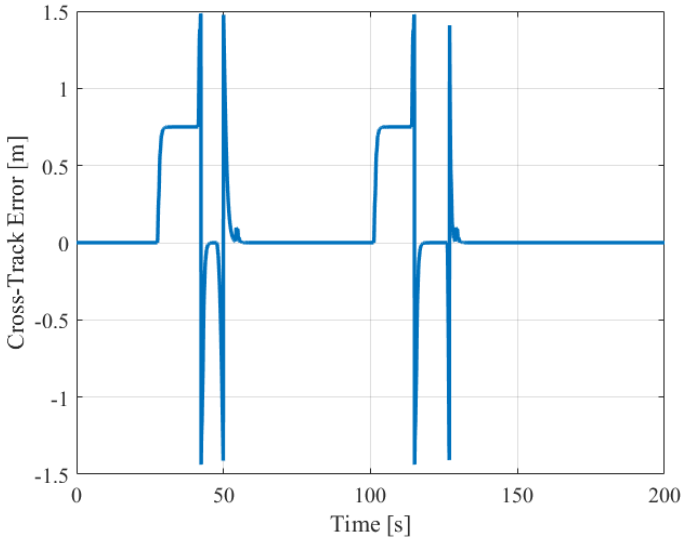

Figure 3.94: Trend of the Cross-Track Error during the Multiple Overtake Scenario

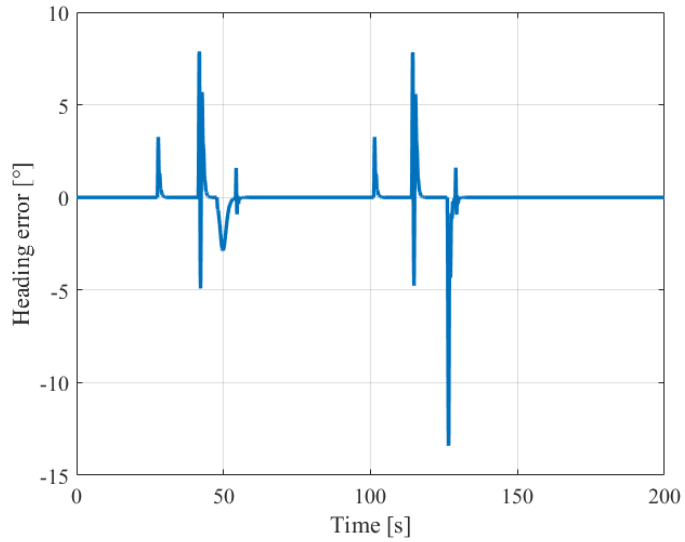

Figure 3.95: Trend of the Heading Error during the Multiple Overtake Scenario

To sum up the Scenario results, we are going to show the Errors as well as the Steering Command, which are in line with the previous Overtake Scenarios.

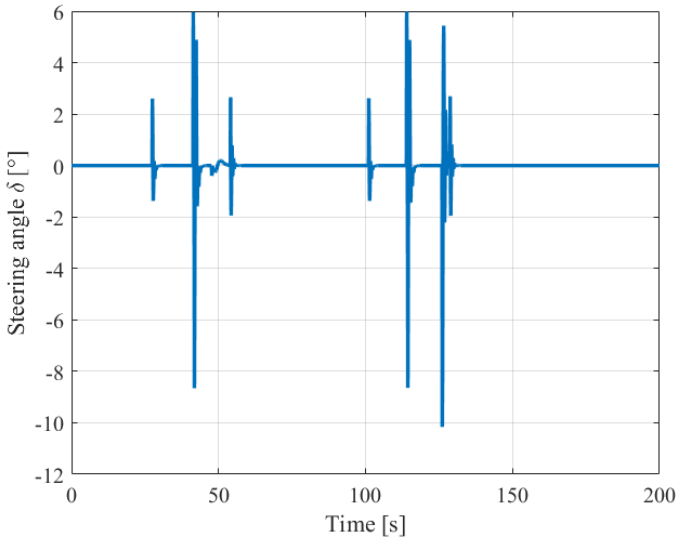

Figure 3.96: Steering Command during the Double Overtake Scenario

3.10 Abort Scenario

We can see that the EgoVehicle indeed enters into the State 5 (ABORT) as can be seen from Figure 3.97, where the curve has a short stint on the 5 value: this short time is coherent with what we expect, given that the Abort maneuver is far simpler in what it does compared to for example the Go Back maneuver does: while the latter requires to move the EgoVehicle laterally for about 3 m and this takes several seconds, up to almost 10 s, the Abort maneuver requires, in general to only correct the Heading of the Vehicle and move it laterally of some cm, since it leads the EgoVehicle back to the "waiting position" i.e. midway between the centerline of the Own Lane and the Own/Overtake Boundary and the EgoVehicle was in the Own Lane still, since otherwise the State Machine would have moved the State to Emergency Overtake.

The heavy steering pikes that can be seen twice in the graph of Figure 3.98 are not due to a mistake in the tuning of the Stanley Controller, but rather to a weakness of the Camera block, which locally is not able to detect any LaneBoundary due to the sharp change of direction of the EgoVehicle: of course such failure is a computational failure due to bottlenecks with Computational Power rather than the inadequacy of our Sensing Equipment, which in real life will have been extensively tested by the producer.

We can see that the Camera failed to provide non-NaN values to our data analysis functions through Figure 3.99, where in two occasions the value of NumLanes drops to -1, which is the default value used to signal that no Lane was detected, as of the

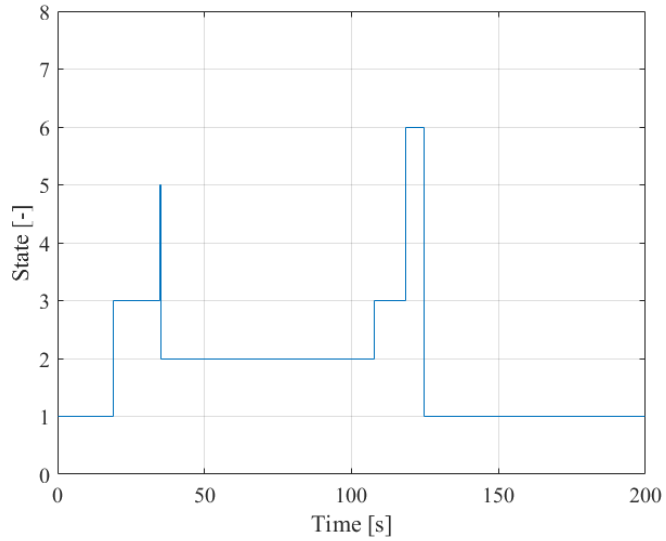

Figure 3.97: Trend of the State during the Abort Scenario

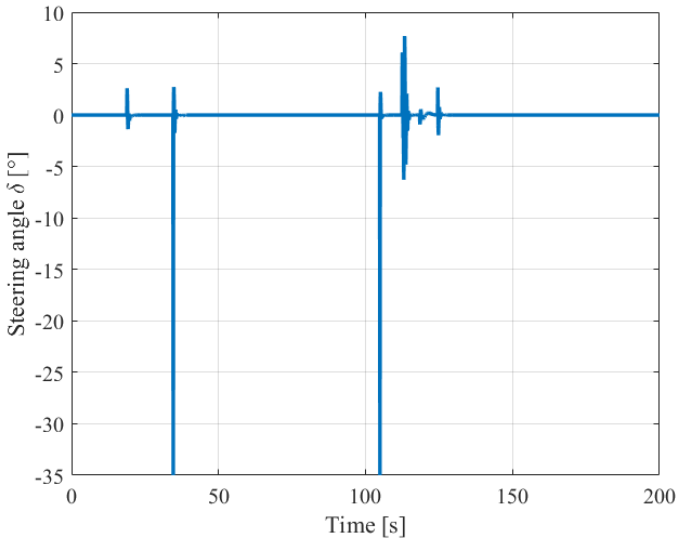

Figure 3.98: Trend of the Steering Angle during the Abort Scenario

Algorithm of Section 2.7.4.

Moreover, in an attempt to show that our System is indeed robust, we want to plot the errors and the trajectory of the Egovehicle as well (Figures 3.100-3.102): we can see that, despite the spikes in the Error values which are locally kicked at 0 in the time instant of the failure of the Camera block, the trajectory of the

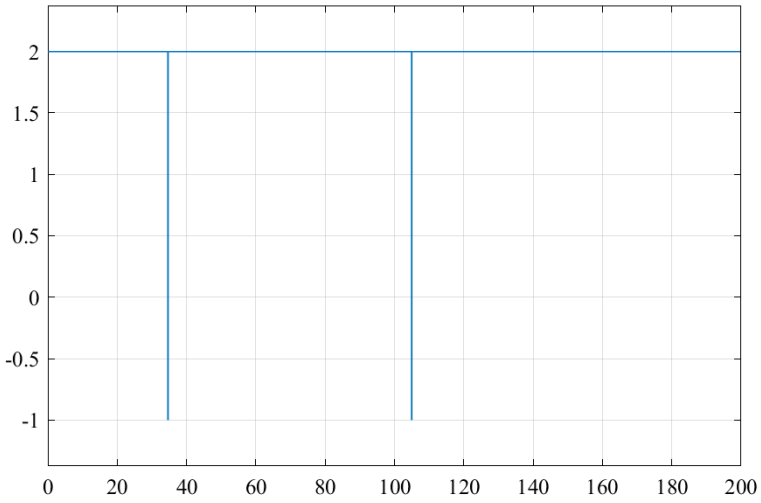

Figure 3.99: Number of the Detected Lanes during the Abort Scenario

EgoVehicle is still acceptable, thus ensuring the comfort of the passengers.

Talking about the comfort of passengers, we also want to have a steady ride,

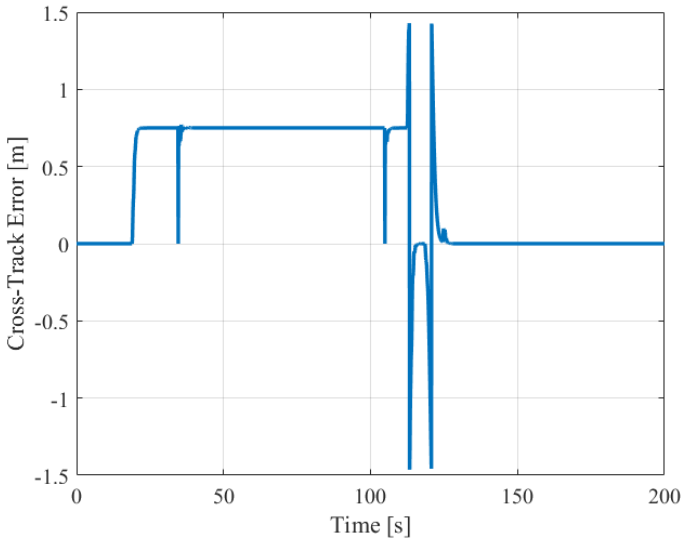

Figure 3.100: Trend of the Cross-Track Error during the Abort Scenario

without a significant longitudinal noise, i.e. without frequent accelerations and decelerations: this was already tested with the ACC_Scenario in Section 3.1 (Figure 3.3 but we want to also ensure it in this more challenging environment:

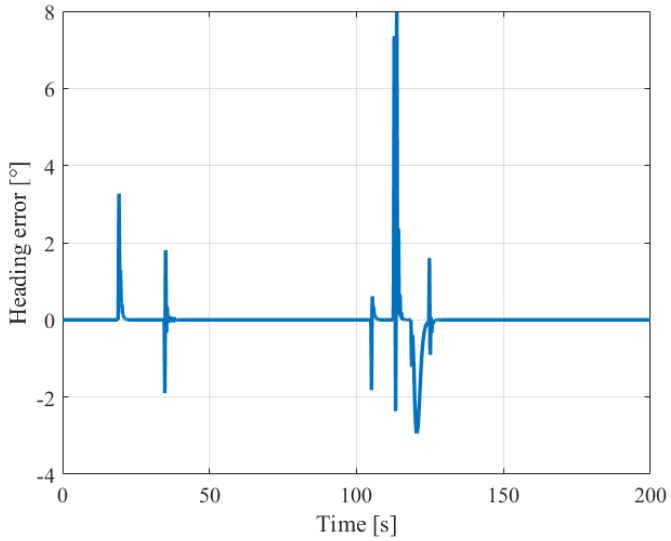

Figure 3.101: Trend of the Heading Error during the Abort Scenario

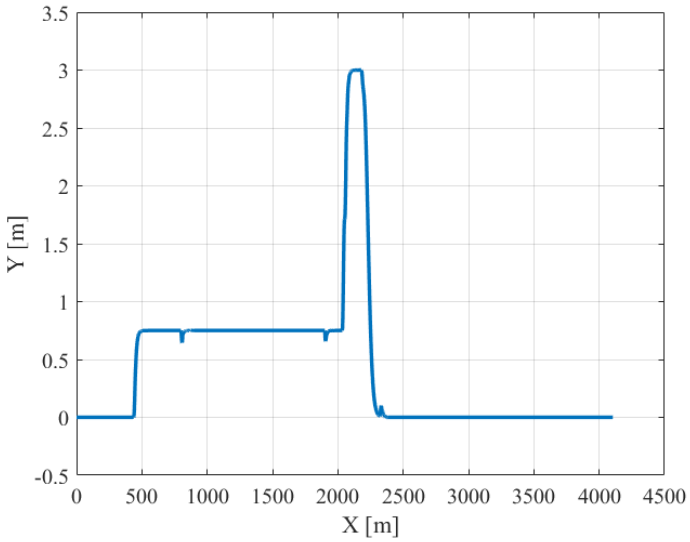

Figure 3.102: Trajectory of the EgoVehicle during the Abort Scenario

from Figure 3.103, we can see that there are some oscillations at the beginning, but their "period" is so long that they are not a nuisance for the passenger; they are anyway due to a limitation on the command, since the first drop is caused by the initial *acc command* being 0 and the second drop has a similar cause, since when we change State (in this case we move from 1 to 3) the PID resets its I component.

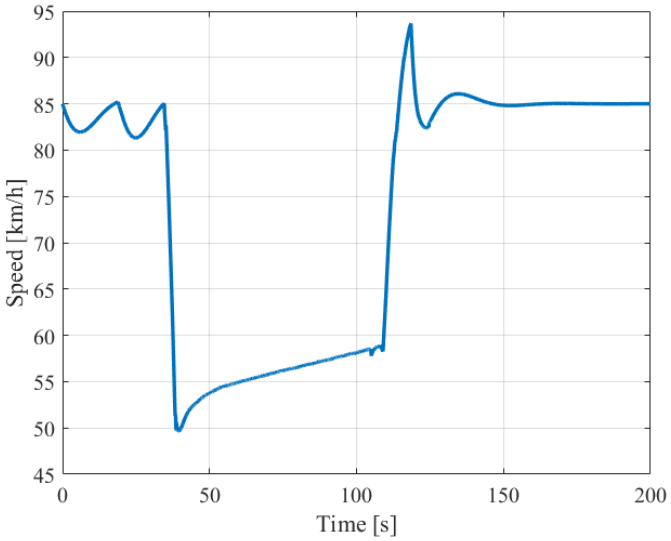

Figure 3.103: Speed of the EgoVehicle during the Abort Scenario

Anyway, as we have said, we do not consider such oscillations to be a harm for the passengers' comfort as they are of small entity and with a long enough period. At the same time, the main deceleration is caused by the aborting of our Overtake: we were running at our desired Speed $VelDes$ (85 km/h) and suddenly we are forced to stay in line behind a slow (between 55 and 60 km/h more or less, according to the info in Section 2.4.10) vehicle as well as to back off from said LEad Vehicle, as we were much closer than our reference distance, as can be seen from Figures 3.104 and 3.105.

Finally, we can see that the Leading Flag is not dropped during the simulation, since the EgoVehicle did not steer enough to lose the Lead Vehicle from its field of view.

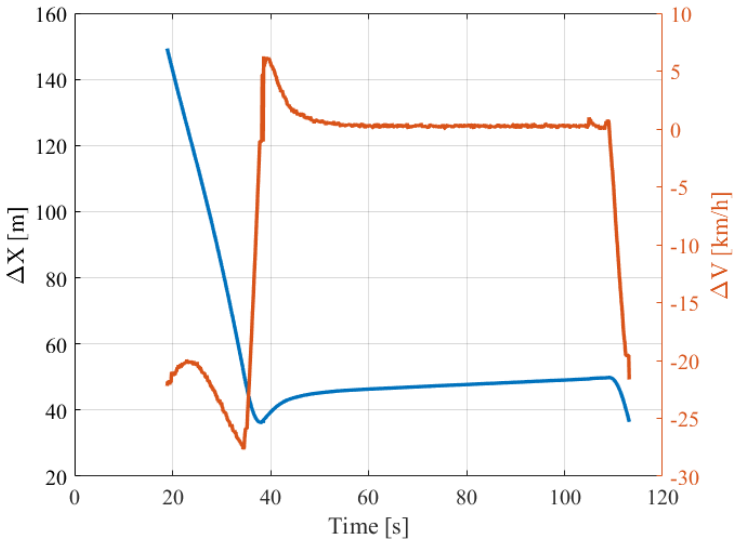

Figure 3.104: Trend of the Radar signals during the Abort Scenario

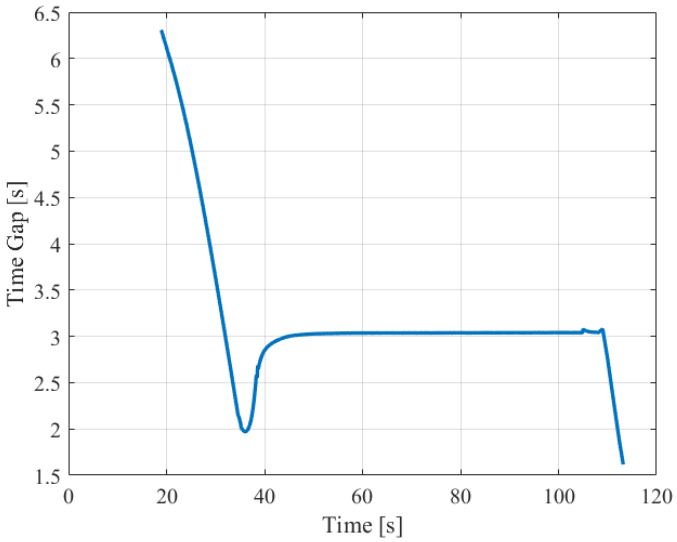

Figure 3.105: Trend of the Time gap during the Abort Scenario

3.11 Emergency Overtake

As we said introducing this Scenario, in this simulation we encounter an Emergency Overtaking, as we can see from Figure 3.107, we indeed move into State 4 (EMERGENCY OVT) during the Overtake Maneuver.

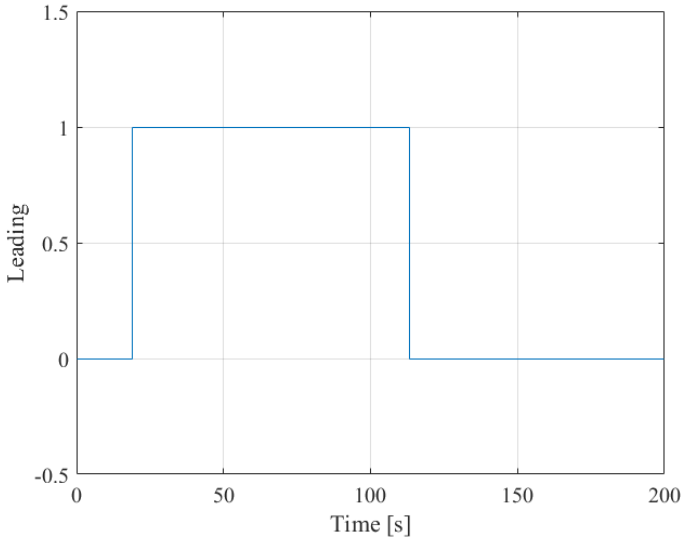

Figure 3.106: Trend of the Leading Flag during the Abort Scenario

Since the purpose and main difference of State 4 compared to basic State 3 is to

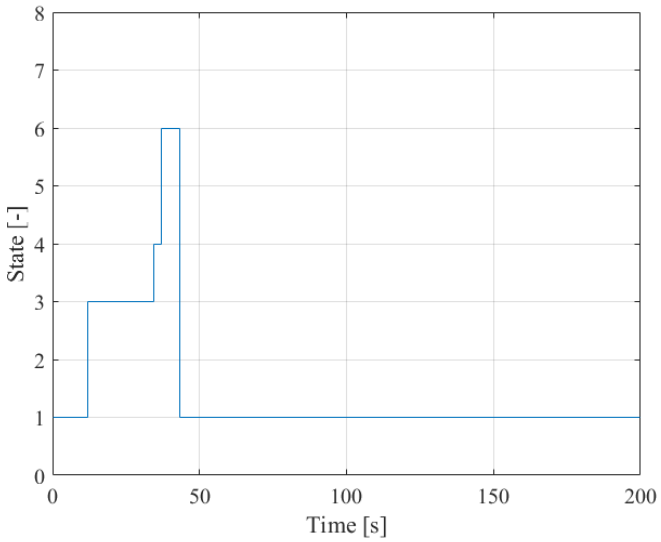

Figure 3.107: Trend of the State during the Emergency Overtake Scenario

get out of the Overtake maneuver as fast as possible, we see a considerable increase in the speed of the EgoVehicle (Figure 3.108).

We can see that the Emergency Ovt was triggered instead of simply Aborting the Overtake because, as we can see from Figure 3.109, when the OvtCounter is raised,

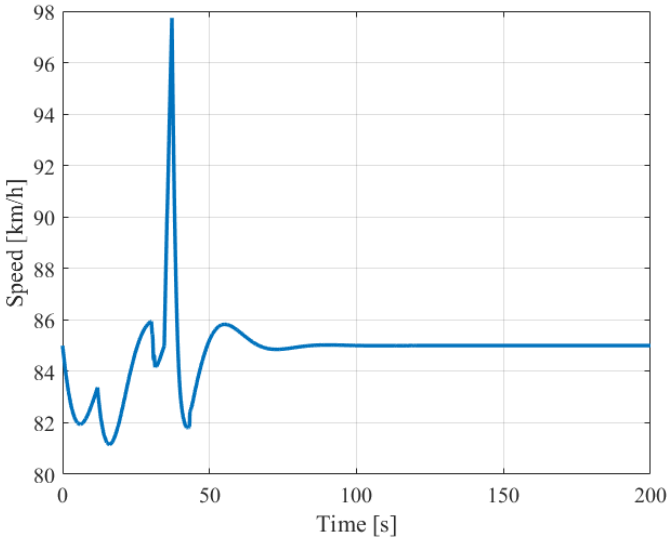

Figure 3.108: Trend of the Speed of the EgoVehicle during the Emergency Overtake Scenario

i.e. at the time instant where the State jumps to State 4 (EMERGENCY OVT), the Vehicle was already in Lane 2, which is consistent with the fact that Radar

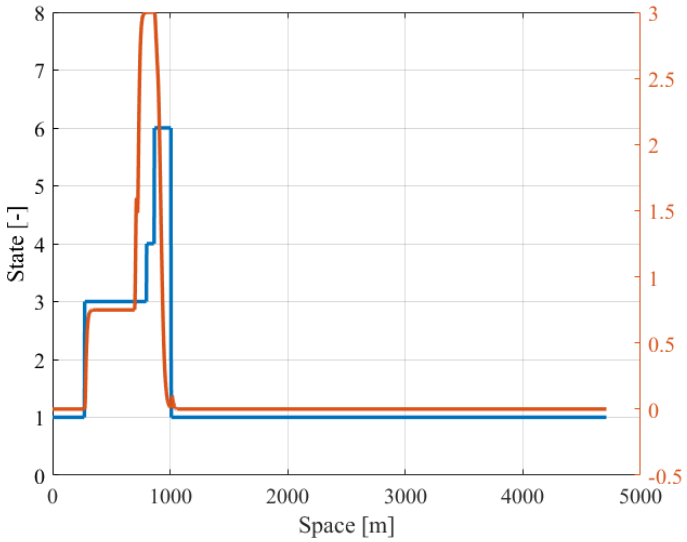

Figure 3.109: State and Trajectory during the Emergency Overtake Scenario

signals and Time Gap disappear around the same time (Figures 3.110 and 3.112).

In fact, we see that the Radar signals end at about the 30 s mark, but the State

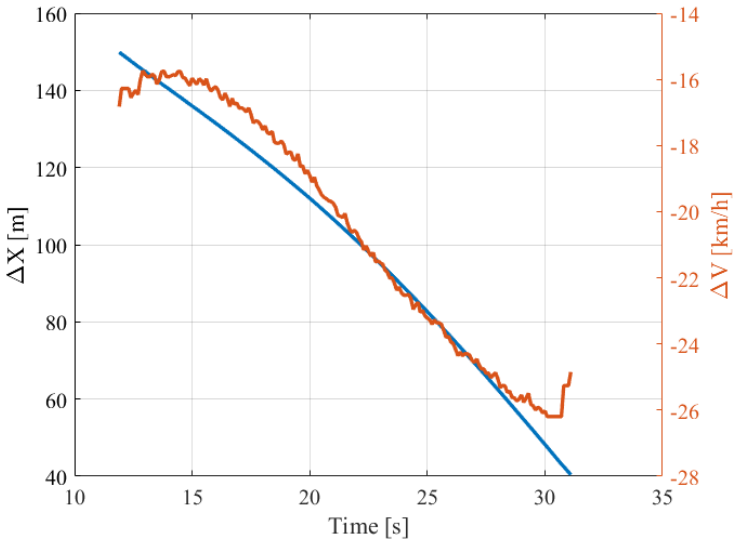

Figure 3.110: Radar signals during the Emergency Overtake Scenario

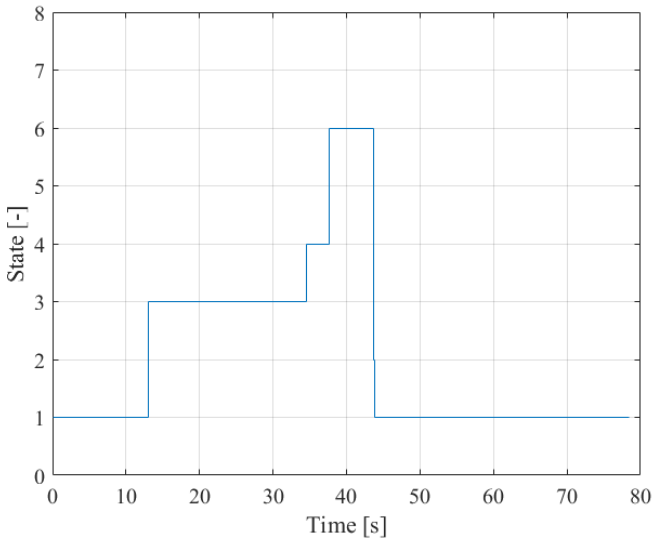

Figure 3.111: Zoom of the State on the first part of the Emergency Overtake Scenario

still is in 3 (OVERTAKE), as can be seen from Figure 3.111, where we see a zoom of the State trend in the first part of the Simulation: this means that when the OvtCounter is raised - because we have met the Continuous Lane Marker - the Lead Vehicle was not output anymore by the Central Radar data analysis (Section

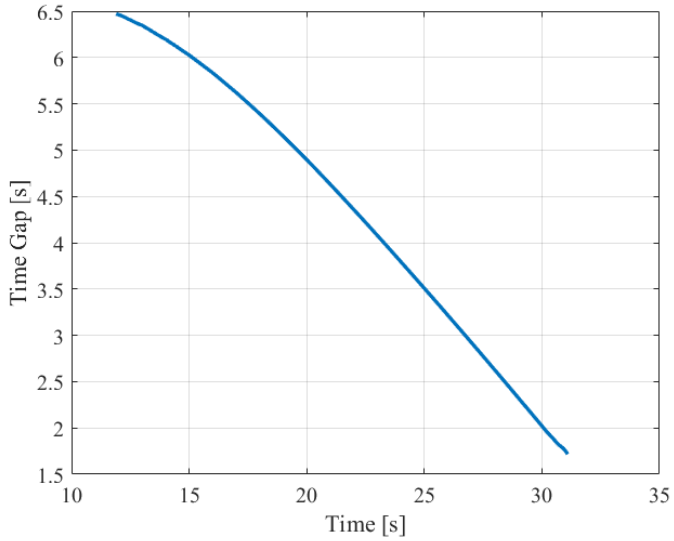

Figure 3.112: Time gap during the Emergency Overtake Scenario

2.7.5 i.e. we were in the Overtake Lane. To sum up the Scenario, we present the Errors and the steering Command, which are not influenced by the Camera failures, as found in Section 3.10, which further prove that the Control and Sensing Systems are overall speaking robust.

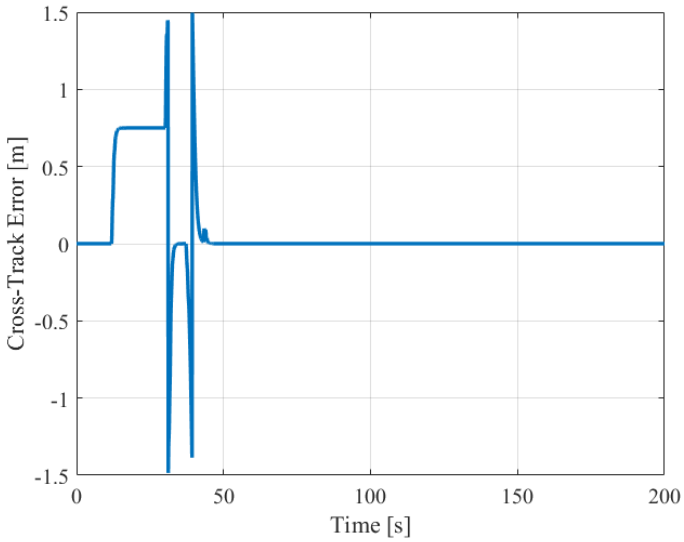

Figure 3.113: Trend of the Cross-Track Error during the Emergency Overtake Scenario

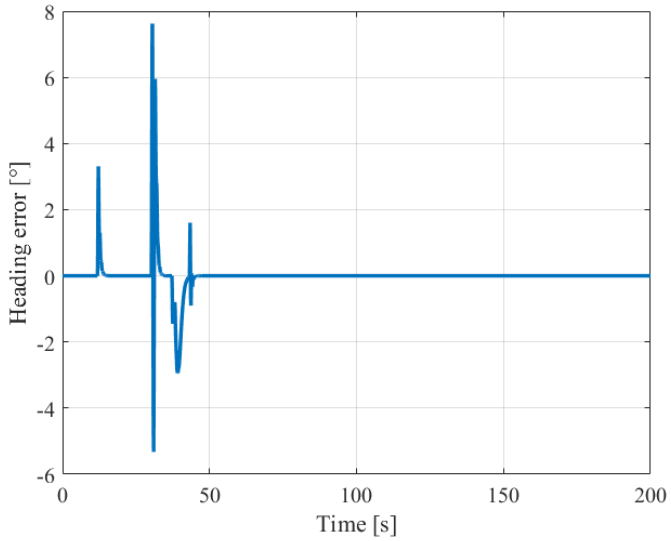

Figure 3.114: Trend of the Heading Error during the Emergency Overtake Scenario

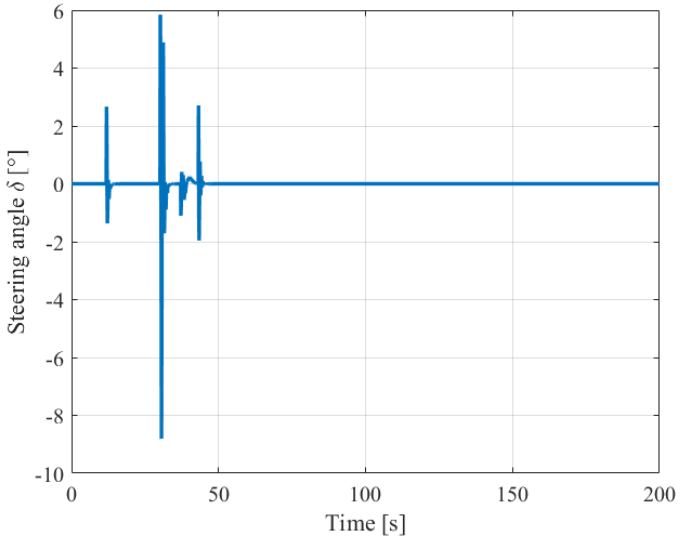

Figure 3.115: Steering Command during the Emergency Overtake Scenario

3.12 Abort Oncoming Vehicle

The first result of this Scenario that we want to comment is of course the Oncoming Flag (Figure 3.116), which - as we would expect - presents two periods with a "1" signal, i.e. we find ourselves in front of an oncoming vehicle twice, coherently with what we described in the presentation of this Scenario.

Through a comparison between Figure 3.116 and Figure 3.117, we can see that

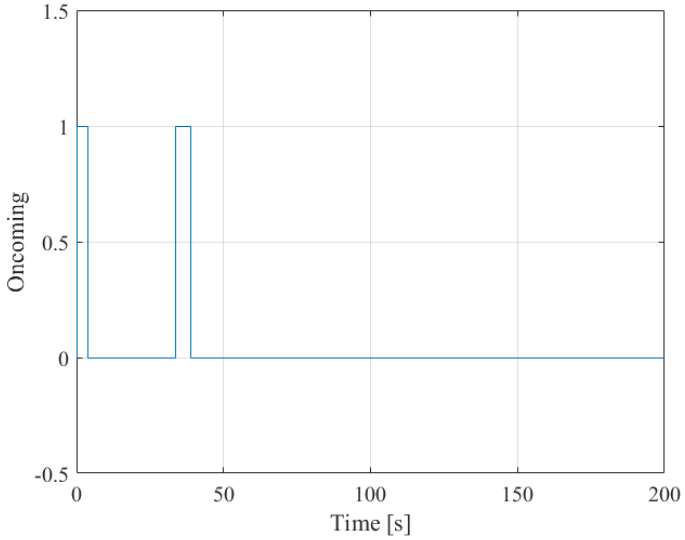

Figure 3.116: Trend of the Oncoming signals during the Abort Oncoming Vehicle Scenario

the State passes to 5 (ABORT) at the same time as the Oncoming Flag gets raised, confirming that we can abort the overtake maneuver if we meet an oncoming vehicle.

However, the EgoVehicle was still well inside the Own Lane, not having begun the proper Overtaking maneuver, but just being in State 3 (OVERTAKE) and this can be noted by looking at Figure 3.118, where there is no significant ripple in correspondence of the ABORT spike (compare with the overlay of Trajectory and State in Figure 3.119), meaning that the EgoVehicle does not need to correct the trajectory, since it did not bend to the left from the "waiting position".

This consideration about the "mild" Abort of the Overtaking maneuver is further reinforced by Figure 3.120, where there are no spikes of the Steering Command δ around the 40 s mark.

The main spikes of the Steering Command in Figure 3.120 are corresponding:

- To the passage from State 1 (STAY) to State 3 (OVERTAKE) immediately

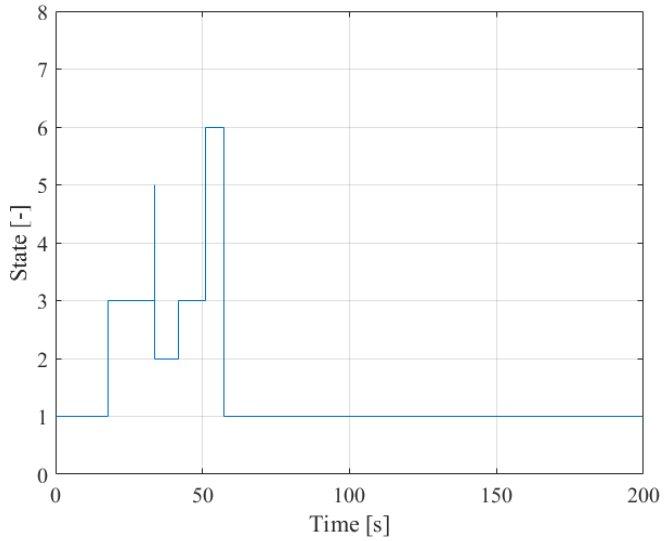

Figure 3.117: Trend of the State during the Abort Oncoming Vehicle Scenario

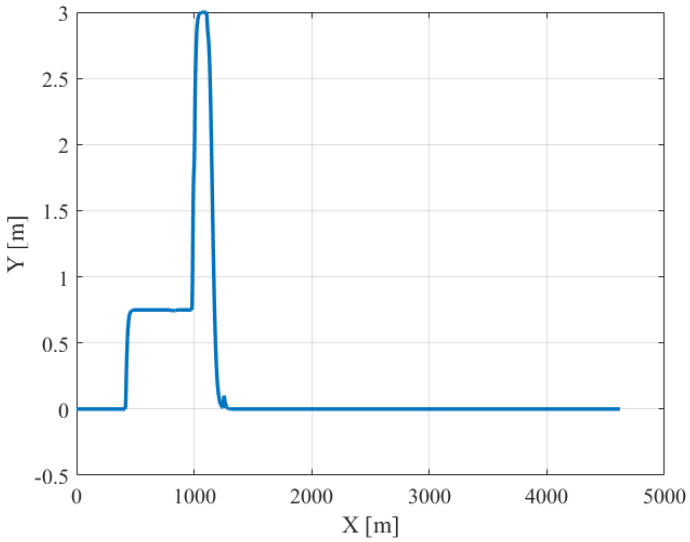

Figure 3.118: Trajectory of the EgoVehicle during the Abort Oncoming Vehicle Scenario

pushing the EgoVehicle into the "waiting position" (around the 20 s mark).

- To the left turn in order to move into the Oncoming Lane so as to overtake the Lead Vehicle (just before the 50 s mark).

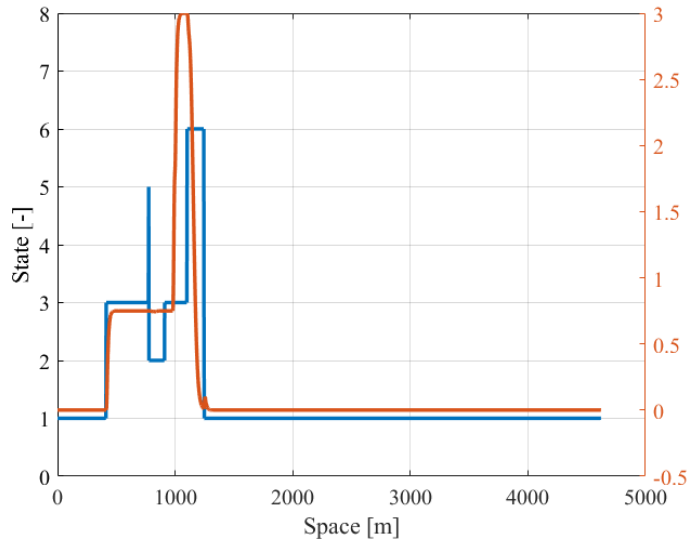

Figure 3.119: State and Trajectory during the Abort Oncoming Vehicle Scenario

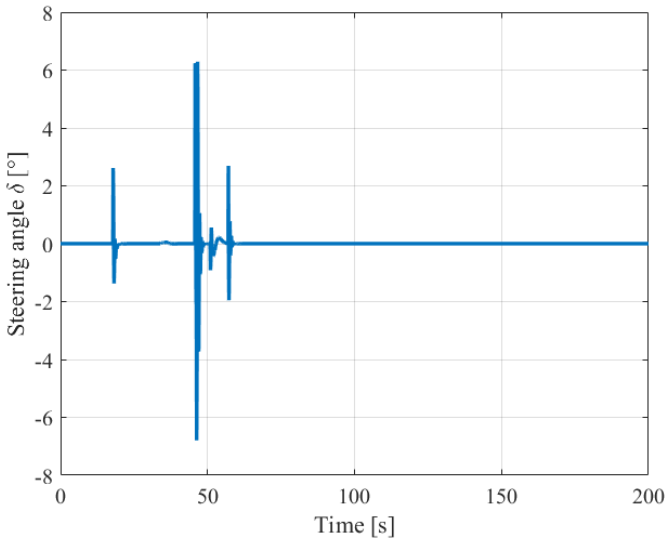

Figure 3.120: Steering Command during the Abort Oncoming Vehicle Scenario

- To the right turn in order to go back to the Own Lane once the Overtake has been completed (after the 50 s mark).

The fact that the EgoVehicle did not steer left aggressively before being forced to Abort is further confirmed by the fact that the Leading Flag never drops (Figure 3.121), since we never steer left enough to lose the Lead Vehicle from the FOV of

our Central Radar; the continuity of the Leading Flag is also coherent with what

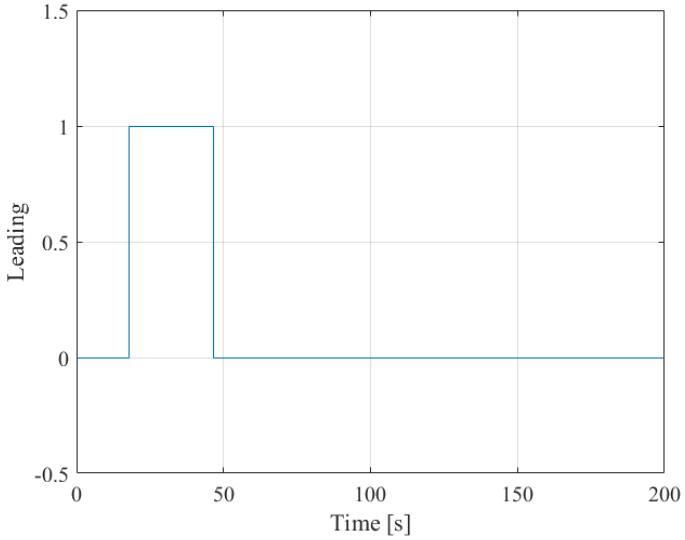

Figure 3.121: Leading Flag during the Abort Oncoming Vehicle Scenario

we see from the Radar signals (Figure 3.122) and Time Gap (Figure 3.123), where we do not see holes due to the loss of tracking of the Lead Vehicle, and the fact that such signals end up around the 50 s mark is confirming the fact that the high Steering Angle peak of Figure 3.120 is the one leading to the Lane Change.

The fact that the Time Gap drops to 2 seconds at the moment of the abortion of the Overtake maneuver explains why the State does not move to State 7 (BRAKE) after the Abort, since we are above the 1.5 seconds threshold set into State 5 for the change into BRAKE State: this means that the recovery of the Safety Distance and the slowing down of the EgoVehicle is performed just by mean of the ACC, i.e. through pure braking on the Front Axle.

3.13 Abort Brake

In this Section we will comment on the results obtained during the **Abort Brake** Scenario, which was the last Scenario we created, in order to put into as much pressure as possible our Decision Making - Path Planning - Control pipeline.

The first result that we want to comment is the trend of the State, which has a passage into State 7 quickly after going back into State 2 (Figure 3.125): The passage to State 7 (BRAKE) after we enter State 2 (WAIT) is caused indeed by the low TTC afer the EgoVehicle goes back to the OwnLane, as is seen from Figure 3.126, where we can see the dip in the TTC at around the same 40 seconds mark

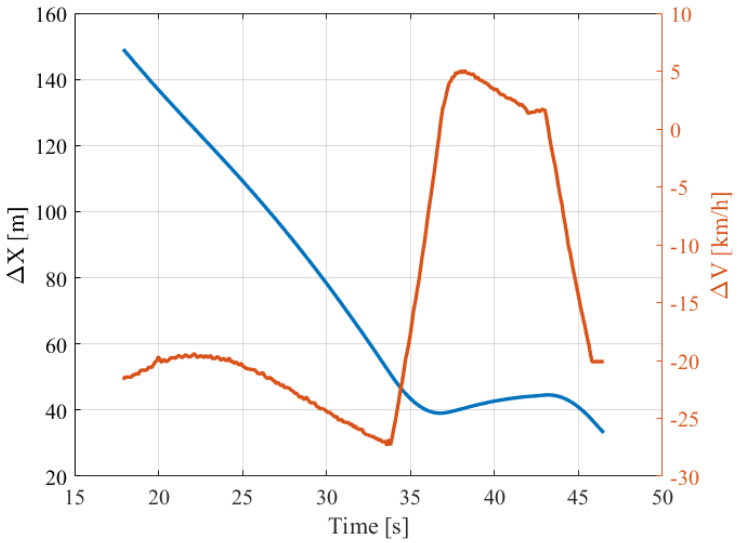

Figure 3.122: Trend of the Radar Signals during the Abort Oncoming Vehicle Scenario

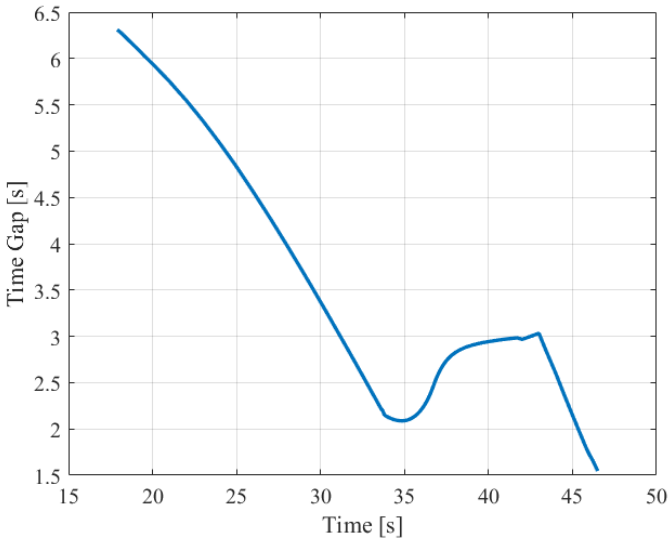

Figure 3.123: Time Gap during the Abort Oncoming Vehicle Scenario

as the fast 3-5-2-1 sequence of States in Figure 3.125 which is where the TTC naturally drops to the lowest value, since we are about to change the Lane and, therefore, we are at the minimum distance from the Lead Vehicle, around 20 m.

What is also very interesting to comment is the gap in the two curves of the

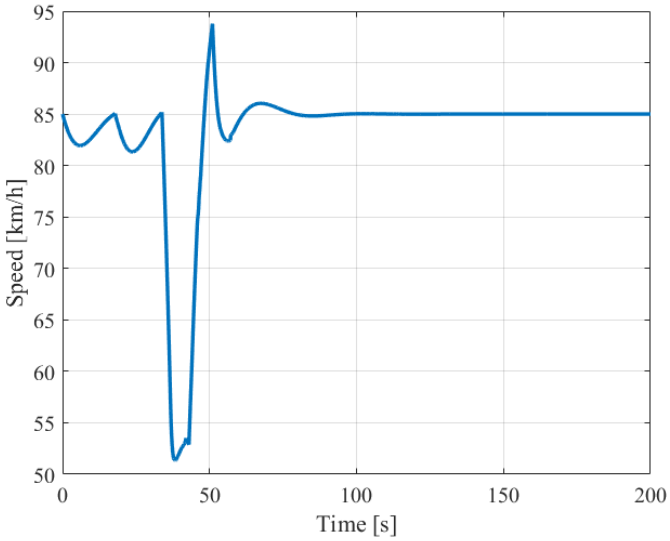

Figure 3.124: Speed of the EgoVehicle during the Abort Oncoming Vehicle Scenario

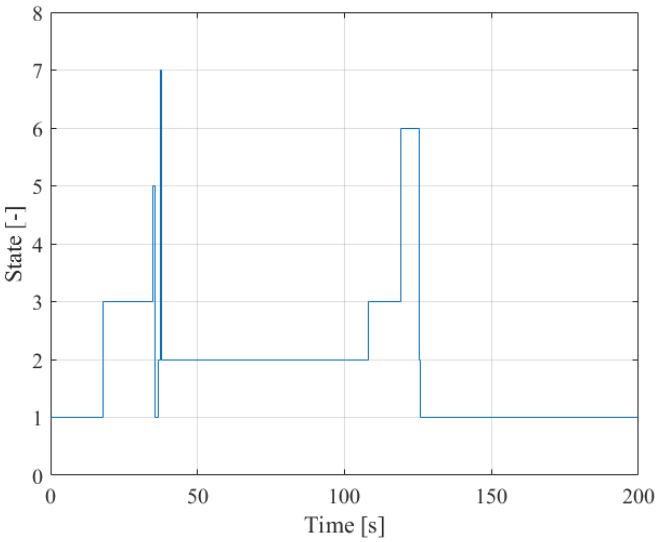

Figure 3.125: Trend of the State during the Abort Brake Scenario

Radar Signals from Figure 3.127: this time, the gap is not caused by a failure of the Camera, but is due to the particular trajectory followed, as of Figure 3.128.

Since the EgoVehicle was already turning left, as highlighted by the high value of the Heading Error e_h in Figure 3.129, which corresponds to the high yaw angle ψ of

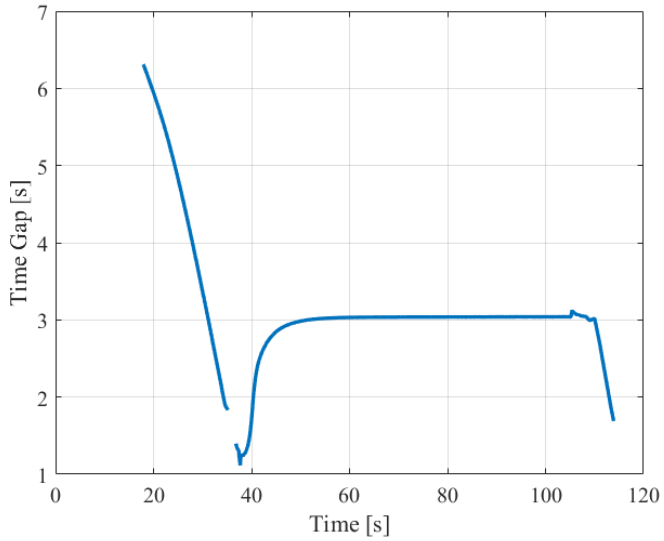

Figure 3.126: Time Gap during the Abort Brake Scenario

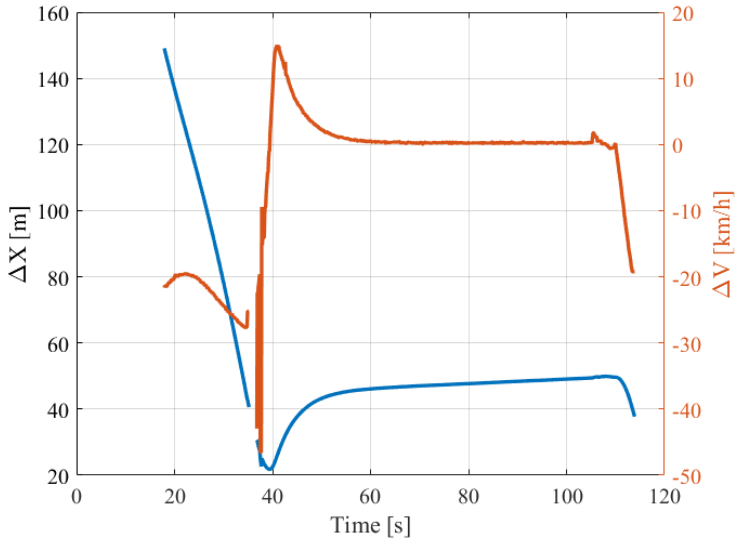

Figure 3.127: Radar Signals during the Abort Brake Scenario

the EgoVehicle at the beginning of the Lane Change maneuver, which is interrupted by the raising of the OvtCounter: the high yaw which needs to be counteracted leads to a strong steering towards the right, i.e. a high negative Steering Angle, as of Figure 3.130.

The fact that we were almost moving to the Oncoming Lane means that the

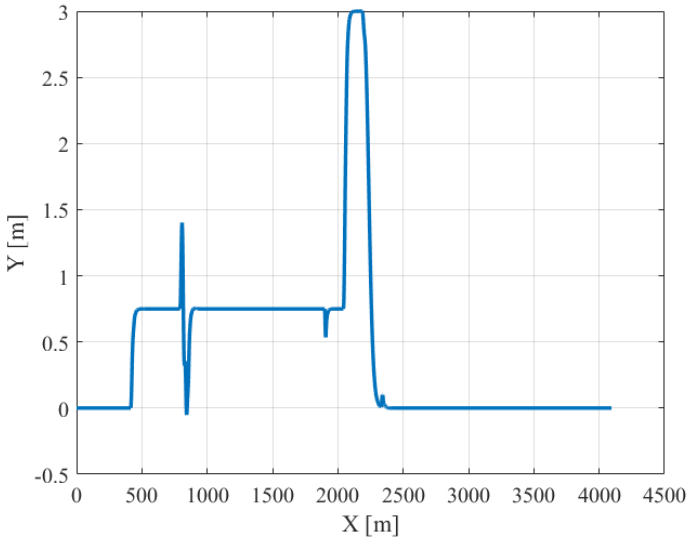

Figure 3.128: Trajectory of the EgoVehicle during the Abort Brake Scenario

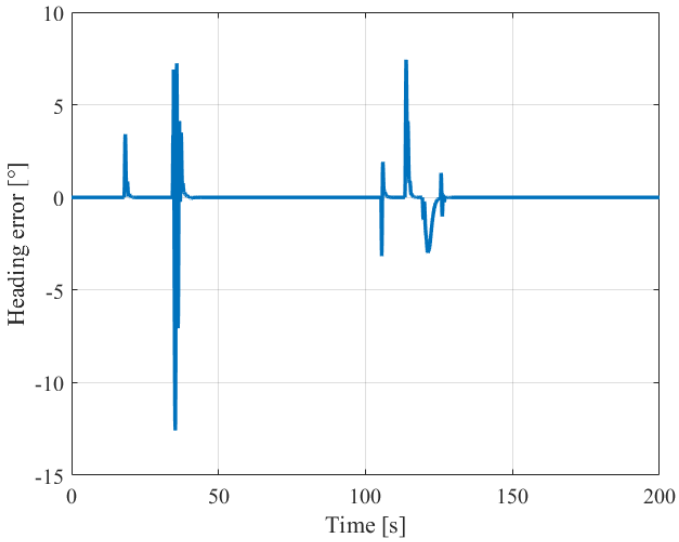

Figure 3.129: Trend of the Heading Error during the Abort Brake Scenario

leading vehicle exits from the FOV of the Central Radar and therefore the Leading Flag drops (Figure 3.131): because of this, when we end the Abort maneuver, we move to State 1 (STAY) in the very next timestep after we entered the State 2 (WAIT) because the GNN Tracker had not yet Tracked the Lead Vehicle; this means that the equilibrium position is not anymore the "waiting position" near

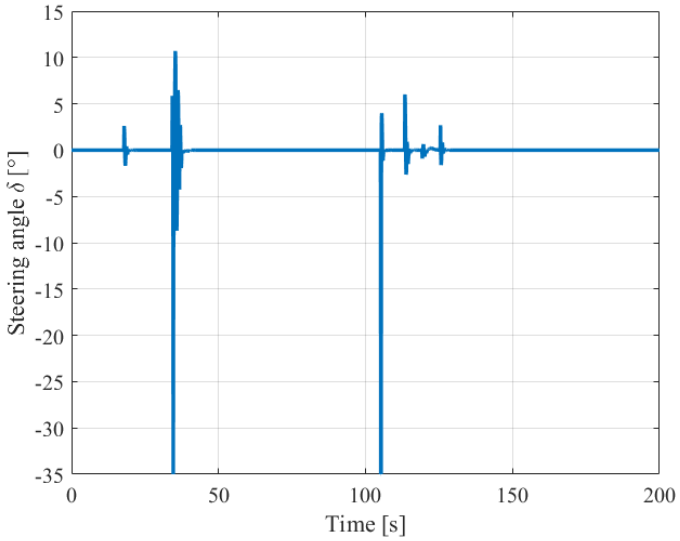

Figure 3.130: Steering Command during the Abort Brake Scenario

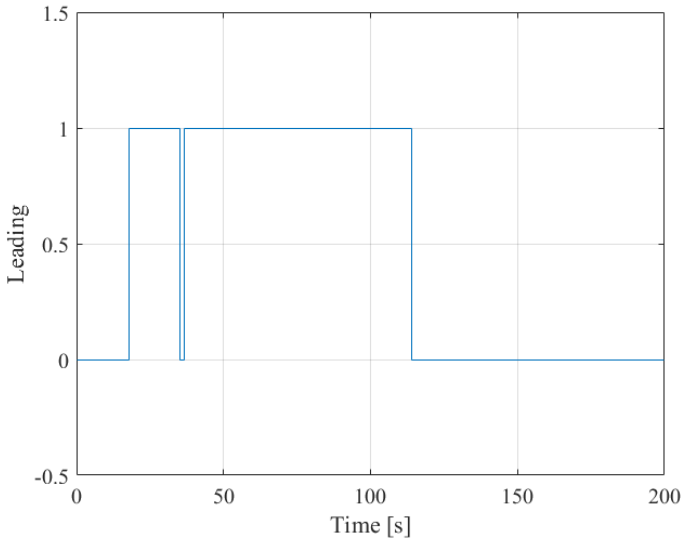

Figure 3.131: Leading Flag during the Abort Brake Scenario

the Lane Marker, but rather the real centerline of the Own Lane. Because of this, the passage in State 1 (STAY) drives the EgoVehicle to having a **nil "real" Cross-Track Error**; in this short time instant in which the EgoVehicle does not have a leading vehicle, the goal speed is 85 km/h and this is evident from Figure 3.132: even though before the 50 s mark the EgoVehicle experiences a severe

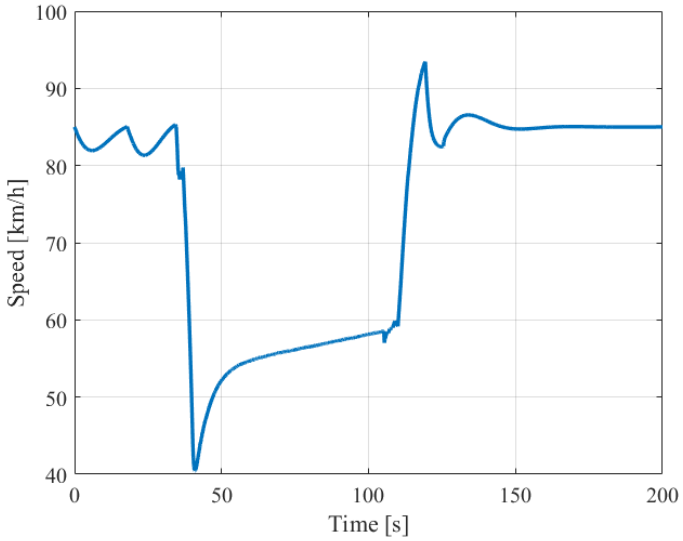

Figure 3.132: Speed of the egoVehicle during the Abort Brake Scenario

braking, if we look more closely at the Speed profile (Figure 3.133), we see that there is an acceleration in correspondence of the gap in the Leading Flag. As soon

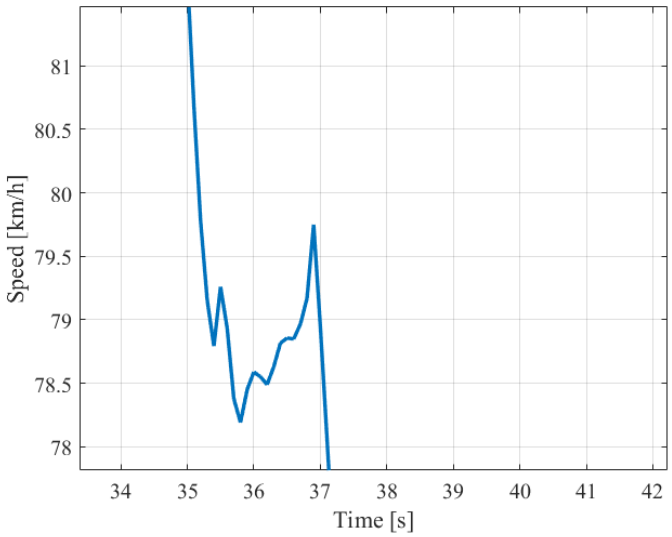

Figure 3.133: Zoom of the Speed of the egoVehicle during the Abort Brake Scenario

as the Leading Flag gets raised again, the EgoVehicle enters into State 7 (BRAKE) and reduces the speed in order to bring itself back to the distance required by the

Constant Time Gap ACC; the following Speed increase is then required in order to return to the correct Vehicle Speed equal to the Lead Vehicle Speed, required by the CTG, which is increasing with the time, as the Lead Vehicle does not have a constant speed: this is highlighted by Figure 3.127, where the relative distance ΔX increases instead of staying constant, while the Time Gap stays constant.

Chapter 4

Analysis and Comments: final remarks and future developments

To sum up this thesis work, we want to have a quick overall comment about the results presented in Section 3; even though results in science need to be validated from 3 different sources, we dare to say ourselves that the results we obtained are satisfactory and, for what concerns us, they validated our Perception, Path Planning and Control Systems, even if - of course - we could improve them significantly, and of course we MUST improve them before we attempt any such test on a Real World vehicle.

4.1 Improvements and Future Works on Perception System

For what concerns the Perception Systems, during the explanation of our Methodology, we have already pointed out that we have in Beta-State the installation of another BlindSpot Radar, this time on the Left Corner, in order to detect if we are being overtaken. We decided, however, not to include it in the main corpus of the thesis work since - because of Computational Power limits - we could not perform enough simulations with dedicated Scenarios with that Radar turned on.
Anyway, below are the parameters of said Radar:

- **Update Rate:** 10 Hz.
- **Sensor mounting relative to Vehicle Origin:** [0 m, 0.9 m, 0.2 m].

- **Sensor rotation relative to Vehicle Frame:** [**135°**, 0°, 0°].
- **Azimuth resolution:** 1°.
- **Range resolution:** 2.5 m.
- **Range rate resolution:** 0.5 m/s.
- **Angular field of view [Azimuth, Elevation]:** [**90°**, 5°].
- **Range Limits:** [0 m, 150 m].
- **Range rate limits:** [-100 m/s 100 m/s].
- **Detection probability:** 0.9.
- **False alarm rate:** 1e-06.

Further on, in a near future we want to integrate much more the various Sensors, so that we do not have "grey zones": an example is if the Lead Vehicle has disappeared from the Central Radar because we have begun the Overtaking maneuver, but it still is in the FoV of the "Overtaken" radar, i.e. the Right BlindSpot Radar. A good example of such ntegration is the one we performed in the block raising the Overtaken Flag, Section 2.7.6
The weakest point of this whole thesis work, in light of the large problems it gave us during the simulations, is the Leading/Oncoming Flag definition: this was particularly fragile when we were overtaking, as we were not parallel to the motion of the Lead Vehicle, leading to noise and imprecisions on the computation of Lead Vehicle speed, with the risk of triggering an unwanted Oncoming Flag. A possible solution to this could be represented by the insertion of a Convolutional Neural Network tasked with the extraction of Features from the Camera images, e.g. the fact that if I see red lights they are likely tail lights, so a leading vehicle, while white lights are the headlights, so they signal an oncoming vehicle; the reason for such omission from this thesis work was once again the limited time, as we would have needed an extensive labelling phase on a huge catalogue of images and a lengthy training for said CNN.
In the same way, we would like to implement a stronger Lane Recognition, possibly GNSS-based, so that for example we know *a priori* the number of Lanes from a database like Google Maps and then perform a Lane Placement through AI knowing already how many Lanes we have; in this way, we would also be able to know if the Overtake Lane is really an Overtake Lane (because we are on a motorway with two lanes per direction) or an Oncoming Lane, and adapt our Decision Making based on this.
Talking about Decision Making improvement, during this thesis work we also

performed some experiments about the decoupling of the Lane Marker and the oncoming vehicles through the creation of two OvtCounters instead of one, but we did not obtain satisfactory results to be inserting them into the final results.

4.2 Improvements and Future Works on Path Planning

As we have alreadt pointed out when discussing the Mountain Road Scenario, the constant $VelDes$ is not a suitable option, as real roads are composed of a mix of straights and curves: because of this, we should be adapting the Cruise Controller so that, instead of a constant preset value, it aims at a value which is an effective obtainable speed, starting from the Curvature of the road as from [91]; such curvature could be obtained not only from the Camera, but also from an integration with GNSS, more similar to what was done in [91].
Moreover, the same considerations can be repeated for the Speed Limit: other than from the maps, it could also be obtained by mean of working on the Camera signals [92], as is already employed in several mass production vehicles, like Audi, BMW, Ford, Mercedes-Benz, Opel and Volkswagen did [93].
Going on with the discussion on the Path Planning future improvements, we want to highlight a factor that was not taken into account in the Sensitivity Analysis of Section 2.1: we assumed, during that Sensitivity Analysis, that the sharpness of the Sigmoid depends only on the speed, but our application should take into account a major difference between the Sigmoid Sensitivity Analysis and its application; in fact, the major factor influencing the sharpness of the Lane Change maneuver in the Scenario simulations was not the speed of the EgoVehicle, but rather the speed of the **Lead Vehicle**, through the ΔV Radar signal: if we imagine to be trying to overtake a Lead Vehicle which is going only 0.5 km/h slower than us, the ΔX coordinate which is the base of the Sigmoid computation would be decreasing much slower, therefore allowing us for a much higher K.
The reason why we did not reperform a second Sensitivity Analysis on the ΔV parameter instead of the V parameter is - once again - **time constraint**. Considering that it took up to 5 minutes to perform a single simulation, if we wanted to perform a sweep on the K value for a single ΔV with the same pitch of K as we did for V, we would need to perform 1000 simulations for a single ΔV, going up to already more than 80 hours, i.e. 3 and a half days if we managed to automatize the code; moreover, the Speed of the Ego Vehicle would also need to be accounted, since $DeltaV$ would be defining the sharpness of the Sigmoid, while the speed V would determine the Vehicle Dynamics and the Lateral forces, therefore turning the *interp1* Matlab function into a proper LUT making the required time explode; a way to cut down the time could be by employing CNN for training, but still it

would have required too much time, also to prepare the training batch.
Finally, in the Path Planning pipeline, we could correct the Sigmoid Curve based on the lateral distance from the Lead Vehicle while we are overtaking it, drwaing it from the Overtaken Radar; in fact, we were already working on some implementations of it, as can be seen by the Block Diagrams, but since it was not even in a Beta-state, it was not included in this thesis work.
Overall, the Path Planning is suboptimal, since - compared to real human-driven vehicles - we turn left too early, therefore exposing ourselves to too great dangers, since we spend longer time in the Oncoming Lane: however, this problem is to be blamed on the Camera, since we tried to turn later, exploiting the whole "slipstream effect" and spending less time in Overtake/Oncoming Lane, but the Control of the Vehicle would start acting bizarrely, since the Lane Detection would undergo problems due to the occlusion by the Lead Vehicle.

4.3 Improvements and Future Works on Controls

Finally, we want to comment on the controls we employed: while the Stanley Controller and the Constant Time Gap ACC are quite refined, the main simplification of our System lies in the total absence of brakes and their representation through the application of the same (negative) torque as the one on the Front Axle on the Rear Axle, just in case we are in States 5 6 or 7, as is seen in Figure 2.13. Of course, in case of a Real Vehicle application, even if just for simple tests, we will need to modify the Longitudinal Controls for all cases, in order to take into account the possibility to enact the brakes on all the four wheels.

4.4 Final Remarks

Overall, we are moderately satisfied with the results of this thesis work: the fact that, by using the experience accumulated during the experience of the 2021/2022 Season Squadra Corse Polito DRIVERLESS for an application to an everyday scenario is obviously making us proud, as well as the fact that we attempted a work on something which would be a great step forward from the current technology framework, since the Highway Chauffeur can not be used on a two-way road and it is still itself in an early stage.
On the other hand, the fact that it was only applied through Simulations and that the high Computational Cost of the whole System (albeit the most comes from the Scenario Reader and the Perception blocks) can not leave us fully satisfied, since the goal is, as always, to find **practical** applications.
Considering both sides of the coin, the total experience of this thesis work is anyway positive and we hope to have laid out the foundations for future works.

Appendix A

Vehicle Specifics

A.1 Vehicle geometrical characteristics

- **Front axle - CG distance** a**:** $1.10m$
- **Rear axle - CG distance** b**:** $1.30m$
- **Wheelbase** L**:** $2.40m$
- **Front track** T_f**:** $1.67m$
- **Rear track** T_r**:** $1.67m$
- **Center of Gravity height** h_{CG}**:** $0.46m$
- **Vehicle mass** m**:** $1350kg$
- **Yaw inertia** J_z**:** $1900kgm^2$

A.2 Aerodynamic characteristics

- **Frontal Area** A_f**:** $1.70m^2$
- **Drag coefficient** C_x**:** 0.32
- **Negative lift coefficient** C_z**:** NEGLECTED
- **Air density** ρ**:** $1.225kg/m^3$
- **Center of Aerodynamic pressure** h_a**:** $0.57m$

A.3 Suspension characteristics

- **Front roll height $H_{roll,f}$:** $0.41m$
- **Rear roll height $H_{roll,r}$:** $0.49m$
- **Center roll height H_{roll}:** $0.45m$
- **Roll inertia J_x:** $490kgm^2$
- **Front axle springs stiffness $K_{s,f}$:** $59780N/m$
- **Front Antiroll-bar torsional stiffness $K_{arb,f}$:** $10000Nm/rad$
- **Front axle torsional damping G_f:** $2400Nm/rad/s$
- **Rear axle springs stiffness $K_{s,r}$:** $38220N/m$
- **Rear Antiroll-bar torsional stiffness $K_{arb,f}$:** $18000Nm/rad$
- **Rear axle torsional damping G_r:** $2400Nm/rad/s$

A.4 Wheel and Tyre characteristics

- **Wheel Inertia I_r:** $0.57kgm^2$
- **Rolling Resistance linear factor f_0:** $0.0041m$
- **Rolling Resistance quadratic factor f_2:** $2.05e-7m/(rad/s)^2$
- **Wheel radial stiffness K_v:** $225000N/m$
- **Front tyre radius R_f:** $0.35m$
- **Rear tyre radius R_r:** $0.35m$
- **Pacejka 1989 coefficients**

i	a_i	b_i
0	1.2	2.37272
1	-34.0	-9.46
2	1600	1000
3	1315.6	1.3
4	1.96	276
5	0.00501	0.4
6	-0.02103	0.00402
7	0.77394	-0.0615
8	0.002289	1.2
9	0.013442	0.0299
10	0.03709	-0.176
11	19.1656	—
12	1.21356	—
13	6.26206	—

- **Transversal Relaxation Length $L_{ril,T}$:** $0.5m$
- **Longitudinal Relaxation Length $L_{ril,T}$:** $0.2m$

A.5 Powertrain characteristics

- **I gear reduction ratio τ_I:** 4
- **II gear reduction ratio τ_{II}:** 2.2
- **III gear reduction ratio τ_{III}:** 1.2
- **Final drive reduction ratio τ_f:** 3.2

Acc Command \ Engine speed [rpm]	0	400	800	1200	1600	2000	2400	2800	3200	3600	4000	4400	4800	5200	5600	6000	6400	6800	7200	7600	8000	8400	8800	9200	9600
100	-200.00	-199.90	-201.20	-196.98	-199.97	-200.12	-197.25	-196.12	-182.60	-167.12	-146.40	-135.10	-124.49	-114.02	-107.47	-97.53	-92.40	-88.42	-82.16	-77.90	-74.63	-70.52	-68.82	-63.54	-61.12
90	-180.00	-179.91	-181.08	-177.28	-179.97	-180.11	-177.53	-176.51	-164.34	-150.41	-131.76	-121.59	-112.05	-102.62	-96.72	-87.78	-83.16	-79.58	-73.95	-70.11	-67.17	-63.47	-61.94	-57.18	-55.01
80	-160.00	-159.92	-160.96	-157.58	-159.98	-160.10	-157.80	-156.90	-146.08	-133.69	-117.12	-108.08	-99.60	-91.21	-85.97	-78.03	-73.92	-70.74	-65.73	-62.32	-59.71	-56.42	-55.06	-50.83	-48.90
70	-140.00	-139.93	-140.84	-137.89	-139.98	-140.08	-138.08	-137.28	-127.82	-116.98	-102.48	-94.57	-87.15	-79.81	-75.23	-68.27	-64.68	-61.90	-57.51	-54.53	-52.24	-49.36	-48.17	-44.48	-42.78
60	-120.00	-119.94	-120.72	-118.19	-119.98	-120.07	-118.35	-117.67	-109.56	-100.27	-87.84	-81.06	-74.70	-68.41	-64.48	-58.52	-55.44	-53.05	-49.30	-46.74	-44.78	-42.31	-41.29	-38.12	-36.67
50	-100.00	-99.95	-100.60	-98.49	-99.99	-100.06	-98.63	-98.06	-91.30	-83.56	-73.20	-67.55	-62.25	-57.01	-53.73	-48.77	-46.20	-44.21	-41.08	-38.95	-37.32	-35.26	-34.41	-31.77	-30.56
40	-80.00	-79.96	-80.48	-78.79	-79.99	-80.05	-78.90	-78.45	-73.04	-66.85	-58.56	-54.04	-49.80	-45.61	-42.99	-39.01	-36.96	-35.37	-32.87	-31.16	-29.85	-28.21	-27.53	-25.42	-24.45
30	-60.00	-59.97	-60.36	-59.09	-59.99	-60.04	-59.18	-58.84	-54.78	-50.14	-43.92	-40.53	-37.35	-34.21	-32.24	-29.26	-27.72	-26.53	-24.65	-23.37	-22.39	-21.16	-20.65	-19.06	-18.34
20	-40.00	-39.98	-40.24	-39.40	-39.99	-40.02	-39.45	-39.22	-36.52	-33.42	-29.28	-27.02	-24.90	-22.80	-21.49	-19.51	-18.48	-17.68	-16.43	-15.58	-14.93	-14.10	-13.76	-12.71	-12.22
10	-20.00	-19.99	-20.12	-19.70	-20.00	-20.01	-19.73	-19.61	-18.26	-16.71	-14.64	-13.51	-12.45	-11.40	-10.75	-9.75	-9.24	-8.84	-8.22	-7.79	-7.46	-7.05	-6.88	-6.35	-6.11
5	-10.00	-10.00	-10.06	-9.85	-10.00	-10.01	-9.86	-9.81	-9.13	-8.36	-7.32	-6.76	-6.22	-5.70	-5.37	-4.88	-4.62	-4.42	-4.11	-3.89	-3.73	-3.53	-3.44	-3.18	-3.06
-5	10.00	10.00	10.06	9.85	10.00	10.01	9.86	9.81	9.13	8.36	7.32	6.76	6.22	5.70	5.37	4.88	4.62	4.42	4.11	3.89	3.73	3.53	3.44	3.18	3.06
-10	20.00	19.99	20.12	19.70	20.00	20.01	19.73	19.61	18.26	16.71	14.64	13.51	12.45	11.40	10.75	9.75	9.24	8.84	8.22	7.79	7.46	7.05	6.88	6.35	6.11
-20	40.00	39.98	40.24	39.40	39.99	40.02	39.45	39.22	36.52	33.42	29.28	27.02	24.90	22.80	21.49	19.51	18.48	17.68	16.43	15.58	14.93	14.10	13.76	12.71	12.22
-30	60.00	59.97	60.36	59.09	59.99	60.04	59.18	58.84	54.78	50.14	43.92	40.53	37.35	34.21	32.24	29.26	27.72	26.53	24.65	23.37	22.39	21.16	20.65	19.06	18.34
-40	80.00	79.96	80.48	78.79	79.99	80.05	78.90	78.45	73.04	66.85	58.56	54.04	49.80	45.61	42.99	39.01	36.96	35.37	32.87	31.16	29.85	28.21	27.53	25.42	24.45
-50	100.00	99.95	100.60	98.49	99.99	100.06	98.63	98.06	91.30	83.56	73.20	67.55	62.25	57.01	53.73	48.77	46.20	44.21	41.08	38.95	37.32	35.26	34.41	31.77	30.56
-60	120.00	119.94	120.72	118.19	119.98	120.07	118.35	117.67	109.56	100.27	87.84	81.06	74.70	68.41	64.48	58.52	55.44	53.05	49.30	46.74	44.78	42.31	41.29	38.12	36.67
-70	140.00	139.93	140.84	137.89	139.98	140.08	138.08	137.28	127.82	116.98	102.48	94.57	87.15	79.81	75.23	68.27	64.68	61.90	57.51	54.53	52.24	49.36	48.17	44.48	42.78
-80	160.00	159.92	160.96	157.58	159.98	160.10	157.80	156.90	146.08	133.69	117.12	108.08	99.60	91.21	85.97	78.03	73.92	70.74	65.73	62.32	59.71	56.42	55.06	50.83	48.90
-90	180.00	179.91	181.08	177.28	179.97	180.11	177.53	176.51	164.34	150.41	131.76	121.59	112.05	102.62	96.72	87.78	83.16	79.58	73.95	70.11	67.17	63.47	61.94	57.18	55.01
-100	200.00	199.90	201.20	196.98	199.97	200.12	197.25	196.12	182.60	167.12	146.40	135.10	124.49	114.02	107.47	97.53	92.40	88.42	82.16	77.90	74.63	70.52	68.82	63.54	61.12

Table A.1: Condensed version of the Motor map

Bibliography

[1] Umberto Eco. *Come si fa una tesi di laurea*. Milano, IT: Bompiani, 1977 (cit. on p. ii).

[2] *Formula SAE Rules*. July 2020, pp. 121–136 (cit. on p. 1).

[3] Juraj Kabzan et al. *AMZ Driverless: The Full Autonomous Racing System*. ETH Zurich (cit. on p. 1).

[4] Albert Boretti. «Changes of E-KERS Rules to Make F1 More Relevant to Road Cars». In: *Advances in Technology Innovation* 3.1 (2018), pp. 26–35 (cit. on p. 1).

[5] *Six Formula 1 innovations you'll find in road cars*. https://www.fleetcare.com.au/news-fleettorque/six-formula-1-innovations-you%E2%80%99ll-find-in-road-cars. Accessed: 2022-07-06 (cit. on p. 1).

[6] *The Everyday Car Tech That Began Life In F1*. https://news.jardinemotors.co.uk/lifestyle/the-everyday-car-tech-that-began-life-in-f1. Accessed: 2022-07-06 (cit. on p. 1).

[7] https://www.squadracorsedriverless.com/. Accessed: 2022-07-07 (cit. on p. 1).

[8] https://www.cars.polito.it/news/squadra_corse_driverless_student_team_has_been_estabilished. Accessed: 2022-07-07. Jan. 16, 2014 (cit. on p. 1).

[9] *SAE International J3016*. https://www.sae.org/standards/content/j3016_201401/. Accessed: 2022-07-07 (cit. on pp. 1, 5).

[10] Laurène Claussmann, Marc Revilloud, Dominique Gruyere, and Sébastien Glaser. «A Review of Motion Planning for Highway Autonomous Driving». In: *IEEE Transactions on Intelligent Transportation Systems* 21.5 (2020), pp. 1826–1848 (cit. on p. 1).

[11] *Mercedes rolls out Level 3 autonomous driving tech in Germany*. https://www.therobotreport.com/mercedes-rolls-out-level-3-autonomous-driving-tech-in-germany/. Accessed: 2022-07-07 (cit. on pp. 1, 5).

[12] *Reported road accidents, vehicles and casualties tables for Great Britain*. https://www.sae.org/standards/content/j3016_201401/. Accessed: 2022-09-23 (cit. on p. 2).

[13] Per Gårder. «Segment characteristics and severity of head-on crashes on two-lane rural highways in Maine». In: *Accident Analysis and Prevention* 38.4 (2006), pp. 652–661 (cit. on p. 3).

[14] Vincent Cantin, Martin Lavallière, Simoneau, and Norman Teasdale. «Mental workload when driving in a simulator: Effects of age and driving complexity». In: *Accident Analysis and Prevention* 41 (2009), pp. 763–771 (cit. on p. 3).

[15] Gerald McGwin Jr. and David B. Brown. «Characteristics of traffic crashes among young, middle-aged, and older drivers». In: *Accident Analysis and Prevention* 31 (1999), pp. 181–198 (cit. on p. 3).

[16] T. W. Planek and R. C. Fowler. «Traffic accident problems and exposure characteristics of the aging driver». In: *Journal of Gerontology* 26 (1971), pp. 224–230 (cit. on p. 3).

[17] *Global Status Report on Road Safety: Time for Action*. World Health Organization, 2009 (cit. on p. 4).

[18] *Global Status Report on Road Safety 2018*. World Health Organization, 2018 (cit. on p. 4).

[19] Adnan K. Shaout and Mohammed-Ameen Jarrah. «Cruise Control Technology Review». In: *Computers & Electrical Engineering* 23.4 (1997), pp. 259–271 (cit. on p. 4).

[20] *Advanced Driver Assistance Systems Past, Present and Future*. Seventh International Computer Engineering Conference (ICENCO'2011). 2011 (cit. on p. 4).

[21] Antonio Mancuso, Andrea Tonoli, Nicola Amati, and Angelo Bonfitto. *Study and implementation of lane detection and lane keeping for autonomous driving vehicles*. M. Eng. thesis. Turin, Italy, Dec. 2018 (cit. on pp. 4, 54).

[22] *Highway Chauffeur: state of the art and future evaluations Implementation scenarios and impact assessment*. International Conference of Electrical and Electronic Technologies for Automotive. 2018 (cit. on p. 5).

[23] *Automated Driving Roadmap Status: final for publication v.7.0*. ERTRAC Working Group Connectivity and Automated Driving, July 2015 (cit. on p. 5).

[24] Ami Woo, Baris Fidan, and William W. Melek. «Localization for Autonomous Driving». In: *Handbook of Position Location: Theory, Practice, and Advances, Second Edition* 1 (2019), pp. 1051–1087 (cit. on p. 6).

[25] *Do car GPS devices cause accidents?* `https://electronics.howstuffworks.com/gadgets/automotive/car-gps-accidents.htm`. Accessed: 2022-07-08 (cit. on p. 6).

[26] National Research Council (U.S.). Committee on the Future of the Global Positioning System. *The Global Positioning System: A Shared National Asset*. Washington, D.C.: National Academy of Public Administration, National Academies Press, 1995 (cit. on p. 6).

[27] Nelson Acosta and Juan Toloza. «Techniques to improve the GPS precision». In: *(IJACSA) International Journal of Advanced Computer Science and Applications* 3.8 (2012), pp. 125–130 (cit. on p. 6).

[28] Lisa L. Arnold and Paul A. Zandbergen. «Positional accuracy of the Wide Area Augmentation System in consumer-grade GPS units». In: *Computers & Geosciences* 37 (2011), pp. 883–892 (cit. on p. 6).

[29] Stefano Feraco, Stefano Favelli, Andrea Tonoli, Angelo Bonfitto, and Nicola Amati. «Localization Method for Autonomous Vehicles with Sensor Fusion Using Extended and Unscented Kalman Filters». In: *SAE Technical Paper 2021-01-5089* 37 (2021) (cit. on p. 6).

[30] Stefano Favelli, Andrea Tonoli, Nicola Amati, and Stefano Feraco. «Robust Localization for an Autonomous Racing Vehicle». M. Eng. thesis. Turin, Italy: Politecnico di Torino, 2021 (cit. on pp. 6, 85).

[31] Kyoungtayek Choi, Jae Kyu Suhr, and Ho Gi Jung. «In-Lane Localization and Ego-Lane Identification Method Based on Highway Lane Endpoints». In: *Journal of Advanced Transportation* (2020) (cit. on p. 7).

[32] *Vehicle Localization Based on Visual Lane Marking and Topological Map Matching*. ICRA 2020 - IEEE International Conference on Robotics and Automation (Paris). 2020 (cit. on p. 7).

[33] *Lane Recognition System for Guiding of Autonomous Vehicle*. IEEE Intelligent Vehicles '92 Symposium (Detroit). 1992 (cit. on pp. 7, 54).

[34] Hyun-Jun Cha, Woo-Hyuk Jeong, and Jong-Chan Kim. «Control-Scheduling Codesign Exploiting Trade-Off between Task Periods and Deadlines». In: *Mobile Information Systems* 2016 (2016). Ed. by Hindawi Publishing Corporation (cit. on p. 7).

[35] Ahmed AbdElmoniem, Ahmed Osama, Mohamed Abdelaziz, and Shady Maged. «A path-tracking algorithm using predictive Stanley lateral controller». In: *International Journal of Advanced Robotic Systems* 17 (Nov. 2020) (cit. on pp. 7, 76, 78).

[36] *Mobile Robot Navigation in Cluttered Environment using Reactive Elliptic Trajectories*. 18th World Congress of The International Federation of Automatic Control (Milano). 2011 (cit. on pp. 7, 8).

[37] Plamen Petrov and Fawzi Nashashibi. «Modeling and Nonlinear Adaptive Control for Autonomous Vehicle Overtaking». In: *IEEE transactions on Intelligent Transport Systems* 15.4 (Aug. 2021), pp. 1643–1656 (cit. on pp. 7, 9, 10).

[38] Sébastien Glaser, Benoit Vanholme, and Saïd Mammar. «Maneuver-Based Trajectory Planning for Highly Autonomous Vehicles on Real Road With Traffic and Driver Interaction». In: *IEEE transactions on Intelligent Transport Systems* 11.3 (Sept. 2010), pp. 589–606 (cit. on pp. 7, 10).

[39] *Motion Planning for an Autonomous Vehicle Driving on Motorways by Using Flatness Properties*. 2010 IEEE International Conference on Control Applications, Part of 2010 IEEE Multi-Conference on Systems and Control (Yokohama). 2010, pp. 908–913 (cit. on pp. 7, 10, 11).

[40] *Trajectory Planning for BERTHA - a Local, Continuous Method*. IEEE Intelligent Vehicles Symposium (IV) (Dearborn (MI)). 2014, pp. 450–457 (cit. on pp. 7, 11).

[41] Jason Hardy and Mark Campbell. «Contingency Planning Over Probabilistic Obstacle Predictions for Autonomous Road Vehicles». In: *IEEE transaction on robotics* 29.4 (Aug. 2013), pp. 913–929 (cit. on pp. 7, 11).

[42] *Field Experiments in Rover Navigation via Model-Based Trajectory Generation and Nonholonomic Motion Planning in State Lattices*. 29th International Symposium on Artificial Intelligence, Robotics, and Automation in Space (i-SAIRAS '08). Feb. 2008 (cit. on p. 8).

[43] *Smooth Obstacle Avoidance Path Planning for Autonomous Vehicles*. 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES). Sept. 2018, pp. 1–6 (cit. on pp. 8, 14–16, 78).

[44] Xujiang Huang, Wenzhe Zhang, and Pu Li. «A Path Planning Method for Vehicle Overtaking Maneuver Using Sigmoid Functions». In: *IFAC-PapersOnLine* 52.8 (2019), pp. 422–427 (cit. on pp. 8, 14, 16, 78, 81–83).

[45] Ehsan Malayjerdi, Raivo Sell, Mohsen Malayjerdi, Andres Udal, and Mauro Bellone. «Practical path planning techniques in overtaking for autonomous shuttles». In: *Journal of Field Robotics* 39 (2022), pp. 410–425 (cit. on pp. 8, 14, 16, 78).

[46] David González, Joshué Pérez, Vicente Milanés, and Fawzi Nashashibi. «A Review of Motion Planning Techniques for Automated Vehicles». In: *IEEE transactions on Intelligent Transport Systems* 17.4 (Apr. 2016), pp. 1135–1145 (cit. on p. 8).

[47] Shilp Dixit, Saber Fallah, Umberto Montanaro, Mehrdad Dianati, Alan Stevens, Francis Mccullogh, and Alexandros Mouzakitis. «Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects». In: *Annual Reviews in Control* 45 (2018), pp. 76–86 (cit. on p. 8).

[48] Mihail Pivtoraiko, Ross A. Knepper, and Alonzo Kelly. *Nonholonomic Mobile Robot Motion Planning in State Lattices*. Tech. rep. CMU-RI-TR-07-15. Carnegie Mellon University, May 2007 (cit. on pp. 11–13).

[49] Jun Han and Claudio Moraga. «The influence of the sigmoid function parameters on the speed of backpropagation learning». In: *[Lecture Notes in Computer Science] From Natural to Artificial Neural Computation* 930 (1995), pp. 195–201 (cit. on p. 14).

[50] *VISUALIZING AND UNDERSTANDING RECURRENT NETWORKS*. International Conference on Learning Representations (San Juan, Puerto Rico). May 2016, pp. 1–12 (cit. on p. 14).

[51] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. «ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks». In: (Apr. 2020) (cit. on p. 14).

[52] K. Basterretxea, J. M. Tarela, and I. del Campo. «Approximation of sigmoid function and the derivative for hardware implementation of artificial neurons». In: *IEE Proceedings - Circuits, Devices and Systems* 151.1 (Feb. 2004), pp. 18–24 (cit. on p. 14).

[53] I. del Campo, R. Finker, J. Echanobe, and K. Basterretxea. «Controlled accuracy approximation ofsigmoid function for efficient FPGA-based implementation of artificial neurons». In: *ELECTRONICS LETTERS* 49.25 (Dec. 2013), pp. 1598–1600 (cit. on p. 14).

[54] `https://www.hyundaimotorgroup.com/story/CONT0000000000033583`. Accessed: 2022-09-02 (cit. on p. 14).

[55] Tibor Mocsári. «Analysis of the Overtaking Behaviour of Motor Vehicle Drivers». In: *Acta Technica Jaurinensis* 2.1 (2009), pp. 97–106 (cit. on p. 14).

[56] Manfred Morari, Carlos E. Garcia, and David M. Prett. «Model predictive control: Theory and practice». In: *IFAC Proceedings Volumes* 21.4 (June 1988), pp. 1–14 (cit. on p. 14).

[57] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. «Constrained model predictive control: Stability and optimality». In: *Automatica* 36.6 (June 2000), pp. 789–814 (cit. on p. 14).

[58] Giancarlo Genta and Lorenzo Morello. «The Automotive Chassis. Volume 2: System Design». In: Berlin: Springer, 2009, pp. 265–278 (cit. on pp. 17, 21–24, 51, 115).

[59] Hans Bastiaan Pacejka. «Tire models for vehicle dynamics analysis». In: *Vehicle Systems Dynamics* 21 (Oct. 1991), pp. 569–571 (cit. on pp. 23, 24, 47, 48).

[60] *Dynamics exploration of a single-track rigid car model with load transfer*. 49th IEEE Conference on Decision and Control (Atlanta, GA). 2010 (cit. on p. 24).

[61] *Vehicle Dynamics Simulation Course*. `https://didattica.polito.it/pls/portal30/gap.pkg_guide.viewGap?p_cod_ins=01USNLO&p_a_acc=2022&p_header=S&p_lang=EN&multi=N`. Accessed: 2022-09-20 (cit. on p. 45).

[62] Predrag D. Mrdija, Nenad L. Miljić, Slobodan J. Popović, and Marko N. Kitanović. «A Method for Quick Estimation of Engine Moment of Inertia based on an Experimental Analysis of Transient Working Process». In: *Thermal Science* 22.3 (2018), pp. 1215–1225 (cit. on p. 45).

[63] Rahul A. Lekurwale and S. G. Tarnekar. «Determination of Moment of Inertia of Electrical Machines Using MATLAB». In: *International Journal of Engineering Research & Technology (IJERT)* 1.10 (Dec. 2012) (cit. on p. 45).

[64] Rajeev Ranjan Kumar, Pritha Guha, and Abhishek Chakraborty. «Comparative assessment and selection of electric vehicle diffusion models: A global outlook». In: *Energy. The International Journal* 238.C (Jan. 2022) (cit. on p. 46).

[65] *Characteristics and investigation of selected manufacturing defects of passenger car tires*. 13th International Scientific Conference on Sustainable, Modern and Safe Transport (TRANSCOM 2019), (Novy Smokovec, Slovakia). 2019 (cit. on p. 47).

[66] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011 (cit. on p. 47).

[67] Giancarlo Genta and Lorenzo Morello. «The Automotive Chassis. Volume 1: Components Design». In: Berlin: Springer, 2009, p. 78 (cit. on pp. 47, 49).

[68] Hans Bastiaan Pacejka. «Tire and vehicle dynamics (2 edition)». In: Troy, MI: SAE International, 2006, p. 22 (cit. on p. 49).

[69] J.P. Maurice and H. B. Pacejka. «Relaxation Length Behaviour of Tyres». In: *Vehicle System Dynamics* 27.1 (1997), pp. 339–342 (cit. on p. 49).

[70] Joga Dharma Setiawan, Mochamad Safarudin, and Amrik Singh. «Modeling, simulation and validation of 14 DOF full vehicle model». In: 2009, pp. 1–6 (cit. on p. 50).

[71] Karthikeyan Manohar, Andrea Tonoli, Nicola Amati, and Angelo Bonfitto. *Path planning with Rapidly-exploring Random Tree for autonomous race car*. M. Eng. thesis. Turin, Italy, 2020 (cit. on p. 50).

[72] Ping Lou and Qing-yuan Zeng. «Formulation of equations of motion of finite element form for vehicle–track–bridge interaction system with two types of vehicle model». In: *International Journal for Numerical Methods in Engineering* 62 (2005), pp. 435–474 (cit. on p. 50).

[73] Tey Jing Yuen, Soong Ming Foong, and Rahizar Ramli. «Optimized suspension kinematic profiles for handling performance using 10-degree-of-freedom vehicle model». In: *Journal of MULTI-BODY DYNAMICS* 228.1 (2013), pp. 82–99 (cit. on p. 50).

[74] *Next generation short range radar (SRR) for automotive applications*. 2012 IEEE Radar Conference (Atlanta GA, USA). 2012 (cit. on p. 53).

[75] Ralph Mende and André Zander. *A Multifunctional Automotive Short Range Radar System* (cit. on p. 53).

[76] *Countries that Drive on the Left 2022*. `https://worldpopulationreview.com/country-rankings/countries-that-drive-on-the-left`. Accessed: 2022-09-18 (cit. on p. 58).

[77] *Global Nearest Neighbor Multi Object Tracker*. `https://www.mathworks.com/help/fusion/ref/globalnearestneighbormultiobjecttracker.html`. Accessed: 2022-09-23 (cit. on p. 61).

[78] Kai Yu, Liang Ji, and Xuegong Zhang. «Kernel Nearest-Neighbor Algorithm». In: *Neural Processing Letters* 15 (2002), pp. 147–156 (cit. on p. 61).

[79] *Britannica, v. governor machine component*. `https://www.britannica.com/technology/governor-machine-component`. Accessed: 2022-09-27 (cit. on p. 73).

[80] *The Sightless Visionary Who Invented Cruise Control*. `https://www.smithsonianmag.com/innovation/sightless-visionary-who-invented-cruise-control-180968418/`. Accessed: 2022-09-27 (cit. on p. 73).

[81] Ralph R. Teetor. «Speed control device for resisting operation of the accelerator». 2519859 A. Aug. 1950 (cit. on p. 73).

[82] *Modelling and controller design for a cruise control system*. 5th International Colloquium on Signal Processing & Its Applications (Kuala Lumpur, MY). 2009 (cit. on p. 73).

[83] *Press Release from Mitsubishi: Mitsubishi Motors Develops New Driver Support System*. https://www.mitsubishi-motors.com/en/corporate/pressrelease/corporate/detail429.html. Accessed: 2022-27-09 (cit. on p. 75).

[84] Junmin Wang and Rajesh Rajamani. «Should Adaptive Cruise-Control Systems be Designed to Maintain a Constant Time Gap Between Vehicles?» In: *IEEE transactions on vehicular technology* 53.5 (Sept. 2004), pp. 1480–1490 (cit. on pp. 75, 106).

[85] *A Time Gap-Based Spacing Policy for Full-Range Car-Following*. 20th International Conference on Intelligent Transportation Systems (ITSC) (Yokohama, Japan). Oct. 2017 (cit. on pp. 75, 106).

[86] *Cross-track control for underactuated autonomous vehicles*. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005 (Sevilla, Spain). 2005 (cit. on p. 76).

[87] *Autonomous Overtaking Decision Making of Driverless Bus Based on Deep Q-learning Method*. 2017 IEEE International Conference on Robotics and Biomimetics (Macau SAR, China). 2017 (cit. on p. 86).

[88] *Art. 143 del Codice della Strada. Posizione dei veicoli sulla carreggiata.* https://www.aci.it/i-servizi/normative/codice-della-strada/titolo-v-norme-di-comportamento/art-143-posizione-dei-veicoli-sulla-carreggiata.html. Accessed: 2022-09-30 (cit. on p. 89).

[89] *Articolo 142 Codice della Strada*. https://www.asaps.it/agg_cds_2013_online/art142.htm. Accessed: 2022-09-02 (cit. on p. 120).

[90] https://it.mathworks.com/help/driving/ref/visiondetectiongenerator.html\protect\protect\unhbox\voidb@x\hbox{#}mw_986ae8f8-7f17-41d9-b7f3-fba36826c546. Accessed: 2022-09-15 (cit. on pp. 120, 121).

[91] Giovanni Scapicchi. «Simulazione prestazione massima per un modello puntiforme di motociclo elettrico». BSC in Engineering thesis. Modena, Italy: Università degli studi di Modena e Reggio Emilia, Dipartimento di Ingegneria *Enzo Ferrari*, 2021 (cit. on pp. 122, 187).

[92] S Varun, Surendra Singh, Sanjeev Kunte R, amuel R D Sudhaker, and Philip Bindu. «A Road Traffic Signal Recognition System based on Template matching employing Tree classifier». In: 2007, pp. 360–365 (cit. on p. 187).

[93] Natalia Kryvinska, Aneta Poniszewska-Maranda, and Michal Gregus. «A Road Traffic Signal Recognition System based on Template matching employing Tree classifier». In: 141. 2018, pp. 64–71 (cit. on p. 187).