Politecnico di Torino
Master of science program in Automotive Engineering

Master Thesis

State Estimation for Formula Student Driverless Applications

Relators
Prof. Nicola Amati
Phd. Stefano Favelli
Phd. Raffaele Manca
Phd. Eugenio Tramacere

Candidate
Davide LEONETTI

December 2023

Acknowledgements

To *my parents*, for always being there, for your constant support, for being my safe haven at every moment, and for always helping me in all my difficult times. This achievement would have been impossible without you.

To *Giuseppe, Michela, Andrea, and Alex*, for sharing joys and challenges during this journey, for always being there for moments of relief and advice, but above all for moments of play and rye-based dinners.

To *Francesca*, for being a part of my life for more than two years, for supporting me (not only) throughout this journey, for never making me doubt us even during evenings on the phone 1000km apart, for all the moments spent together, for the ice creams in the center of Turin, for all the photos you make me take, and for everything that is to come.

To *grandpa Vito, grandma Maria, and aunt Ida*, for your love towards me, for your direct and indirect support throughout my journey, and for always being proud of me in words and actions.

To *grandma Rosa and uncle Saverio*, for how proud and happy you would have been today.

To *all my uncles, cousins, and relatives*, for your constant affection.

To *all the members of the Squadra Corse Driverless team*, for giving me this precious opportunity to work in the team and for helping me grow in every way, and to the advisors *Raffaele Manca, Eugenio Tramacere, and Stefano Favelli* for assisting and guiding me in this thesis work and for helping me grow professionally.

To *all my lifelong friends*, for always being there throughout these 5 years, for giving me happy moments in outings in Turin, pizzas, and game nights on the PlayStation at home, and in Andria with long walks and coffee at Bar de Lucia.

To *all the Mirafiori friends*, for trying in every way to slow down this academic journey with endless table football matches and "long" coffees at the vending machines during study breaks.

Ringraziamenti

A *i miei genitori*, per essere stati sempre presenti, per il vostro sostegno costante e per essere stati il mio porto sicuro in ogni momento e per avermi sempre aiutato in tutti i miei momenti di difficoltà. Questo traguardo sarebbe stato impossibile senza di voi.

A *Giuseppe, Michela, Andrea e Alex*,per aver condiviso gioie e sfide durante questo percorso, per esserci sempre stati sia per momenti di sfogo e di consigli, ma soprattutto momenti di gioco e cene a base di segale.

A *Francesca*,per essere da più di due anni parte della mia vita, per avermi supportato (non solo) in tutto questo percorso, per non avermi mai fatto dubitare di noi anche nelle serate a telefono lontani 1000km, per tutti i momenti passati insieme, per i gelati in centro a Torino, per tutte le foto che mi obblighi a scattare e per tutto quello che verrà.

A *nonno Vito,nonna Maria e zia Ida*,per il vostro amore nei miei confronti, per il vostro supporto diretto e indiretto in tutto questo mio percorso e per essere stati sempre a parole e nei gesti orgogliosi di me.

A *nonna Rosa e zio Saverio*, per quanto sareste stati orgogliosi di me e felici oggi.

A *tutti i miei zii, cugini e parenti*,per il vostro costante affetto.

A *tutti i ragazzi del team di Squadra Corse Driverless*,per avermi regalato questa preziosa oppurtunità di lavorare nel team e avermi fatto crescere sotto tutti i punti di vista e a i relatori *Raffaele Manca, Eugenio Tramacere e Stefano Favelli* per avermi aiutato e guidato in questo lavoro di tesi e per avermi fatto crescere dal punto di vista formativo.

A *tutti gli amici di sempre*,per esserci stati sempre in tutti questi 5 anni, per aver regalato momenti felici uscite a Torino, pizze e partite alla Play in casa e ad Andria con lunghe camminate e caffè al Bar de Lucia.

A *tutti gli amici di Mirafiori*,per aver provato in tutti i modi di rallentare questo percorso di studi con infinite partite al biliardino e caffè "lunghi" alle macchinette durante lo studio.

Abstract

The "Formula Student Driverless" is a specific category within the Formula Student competition in which students design, build, and race single-seat vehicles without drivers. This category focuses on the development of autonomous or driverless vehicles, using advanced technologies such as autonomous driving systems, sensors, artificial intelligence, and other emerging technologies to enable the vehicles to operate without human intervention.

In this thesis work carried out in collaboration with the "Squadra Corse Driverless Polito" team, a method for estimating the longitudinal and lateral velocity of the vehicle in the absence of sensors such as a Ground Speed Sensor is introduced and described. Knowledge of these state variables is crucial for managing the controls of an autonomous vehicle. In fact, in addition to enabling basic controls such as Traction Control and Torque Vectoring, knowledge of these variables is essential for improving the mapping of the vehicle's position on the track, which cannot rely solely on GPS sensors.

This work details the process of estimating longitudinal velocity, followed by an explanation of the techniques employed for lateral velocity estimation. After analyzing the most commonly used methods in literature and in traditional vehicles, it describes the efforts undertaken by a Formula Sae team in estimating these variables. Particularly, an algorithm for longitudinal velocity estimation and two different types of Kalman Filters for lateral velocity estimation were designed (one kinematic and one dynamic). This paper elucidates their implementations, functionalities, and introduces an innovative method for merging the outcomes of these two filters.

The entire study underwent extensive validation through numerous simulations, utilizing various models and software, before being further validated on the track with experimental data. Finally, an analysis of the influence of the accuracy of state estimation on the vehicle's odometry was conducted by comparing results obtained on the track using a Ground Speed Sensor with results obtained through estimation.

Sommario

La "Formula Student Driverless" è una categoria specifica all'interno della competizione Formula Student in cui gli studenti progettano, costruiscono e gareggiano con vetture monoposto senza pilota. Questa categoria si concentra sulla realizzazione di veicoli autonomi o senza conducente, utilizzando tecnologie avanzate come sistemi di guida autonoma, sensori, intelligenza artificiale e altre tecnologie emergenti per consentire alle vetture di operare senza l'intervento di un pilota umano.

In questo lavoro di tesi svolto in collaborazione con il team "Squadra Corse Driverless Polito" viene introdotto un metodo per la stima della velocità longitudinale e laterale del veicolo in assenza di sensori quali un Ground Speed Sensor. La conoscenza di queste variabili di stato è fondamentale per la gestione dei controlli di un veicolo a guida autonoma. Infatti, oltre che per l'attivazione di controlli base quali Traction Control e Torque Vectoring, la conoscenza di queste variabili è fondamentale per il miglioramento della mappatura della posizione del veicolo in pista che non può essere basata solo grazie ai sensori GPS.

Questo lavoro spiega nel dettaglio come avviene la stima della velocità longitudinale, per poi descrivere le tecniche adoperate per la stima della velocità laterale. Dopo un'analisi delle tecniche più utilizzate in letteratura e nei veicoli tradizionali viene descritto il lavoro svolto in un team di Formula Sae per la stima di queste variabili. In particolare un algoritmo per la stima della velocità longitudinale e due diversi tipi di Filtri di Kalman per la stima della velocità laterale sono stati progettati (uno cinematico e uno dinamico). In questo paper vengono descritti le loro implementazioni e il loro funzionamento e inoltre viene descritto un metodo innovativo per fondere i risultati dei due filtri.

Tutto il lavoro è stato prima validato attraverso numerose simulazioni, utilizzando diversi modelli e software per poi essere validato anche in pista con dati sperimentali.
Infine un'analisi sull'influenza dell'accuratezza della stima degli stati sull'odometria del veicolo è stata effettuata, confrontando risultati ottenuti in pista con l'ausilio di un Ground Speed Sensor con i risultati ottenuti tramite l'Estimation.

List of Acronyms

- **1C**: Combustion Class
- **1D**: Driverless Class
- **1E**: Electric Class
- **ABS**: Antilock Braking System
- **ACU**: Autonomous Control Unit
- **CC**: Cruise Control
- **CoG**: Center of Gravity
- **ECU**: Electronic Control Unit
- **EKF**: Extended Kalman Filter
- **FoV**: Field of View
- **FWD**: Front-Wheel Drive
- **GNSS**: Global Navigation Satellite System
- **GSS**: Ground Speed Sensor
- **GPS**: Global Positioning System
- **IMU**: Inertial Measurement Unit
- **LTI**: Linear Time-Invariant
- **LKA**: Lane Keeping Assist
- **MPC**: Model Predictive Control
- **NN**: Neural Network
- **PCA**: Principal Component Analysis
- **PI**: Proportional Integral
- **RMS**: Root Mean Square
- **RWD**: Rear-Wheel Drive
- **SLAM**: Simultaneous Localization and Mapping
- **SSI**: Steady State Index

- **TC**: Traction Control
- **TV**: Torque Vectoring
- **TVC**: Torque Vectoring Controlled

List of Symbols

- α: Tire Sideslip Angle
- β: Vehicle Sideslip Angle
- δ: Steering angle (can be referred to steering or wheel)
- $\dot{\psi}$: Yaw angular speed
- γ:Camber angle
- ℓ: wheelbase
- l_F: Distance between front axle and Vehicle Center of Gravity
- l_R: Distance between rear axle and Vehicle Center of Gravity
- s: tire longitudinal slip
- τ: time constant (in transfer function)
- T_f: Front Track
- u: Longitudinal Component of Velocity
- u_k: External Moment Applied to Vehicle
- v: Lateral Component of Velocity
- ω: Angular Speed
- V_x: Velocity along x axis
- V_y: Velocity along y axis
- ϕ: Angle ϕ rotation respect Y axis
- θ: Angle θ rotation respect X axis
- a_x: Acceleration along the x-axis
- a_y: Acceleration along the y-axis
- f_l:referred to front left tire
- f_r: referred to front right tire
- g: Gravity acceleration
- J_z: Moment of inertia about the z-axis

- M_z:External Moment applied
- m: Vehicle Mass
- r_l: referred to rear left tire
- r_r: referred to rear right tire
- ρ: Air Density
- S: Vehicle front Surface
- C_x: Aerodynamic Coefficient X component
- C_y: Aerodynamic coefficient Y component
- F_x: Force exchanged in longitudinal direction
- F_y: Force exchanged in lateral acceleration
- K_{dyn}: Dynamic weight in Combined Filter
- K_{kin}: Kinetic weight in Combined Filter

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Formula Sae and Categories

Formula SAE [1]is an international university competition in engineering design organized by the Society of Automotive Engineers (SAE). It involves the design and production of a formula-style race car, which is evaluated through a series of tests based on its design qualities and engineering efficiency.

Established in 1981, the competition aims to provide university students with a platform to apply the knowledge they have gained during their academic journey. By participating in Formula SAE, students get the opportunity to put their theoretical learning into practical application, gaining invaluable hands-on experience in the field of automotive engineering.

Teams are judged not only on their car's performance but also on their engineering design, cost analysis, and business presentation skills. This aspect adds a layer of real-world applicability, as teams must consider budget constraints and demonstrate their ability to communicate effectively and present their engineering solutions to a hypothetical panel of investors or potential clients.

Currently, the categories of each competition are three: Class 1C (for gasoline-powered vehicles), Class 1E (for electric vehicles), and Class 1D (for Driverless vehicles). In addition to these, a fourth category, called Class 3, is added, in which teams participate by presenting only the design of the vehicle, and do not take part in the static and dynamic events.
The team "Squadra Corse Driverless" of Politecnico di Torino competes in class 1D and participates in the season 2022/23 at the events of Formula Ata and Formula East.

1.2 Driverless Competition

Class 1D is a category within the Formula SAE (Society of Automotive Engineers) competition that focuses on the development of autonomous vehicles designed and built by university students. In this competition, teams are challenged to design and construct an autonomous vehicle capable of completing a series of dynamic events and specific tasks without any human intervention during the runs.

In Formula SAE competitions, points are awarded to teams based on their performance in various static and dynamic events. In Class 1D the points are given in the following way:

Static Events (375pt):

1. Design Event (200pt): In the design event, teams present their vehicle's engineering design and answer questions from a panel of judges. Points are awarded based on the design's innovation, analysis, manufacturability, and overall quality.

2. Cost Event (100pt): In the cost event, teams prepare a report detailing the cost of manufacturing their race car. The judges assess the cost report, and points are given based on the team's cost-conscious decisions and ability to meet budget constraints.

3. Business Presentation (75pt): This event evaluates the team's business and marketing skills. Teams present a business plan for the production and marketing of their vehicle. Points are awarded for the clarity and persuasiveness of the presentation.

Dynamic Events (600pt):

1. Acceleration Event (100pt): The acceleration event assesses the vehicle's acceleration performance in a straight line. Points are awarded based on the car's acceleration time over 75m.

2. Skid Pad Event (100pt): The test assesses the car's cornering ability. The track resembles the figure of an 8 with two circles of 15.25 meters in diameter, with the starting/finishing line placed at the center and delimited by sixteen cones on the outer side and sixteen cones along the inner side of each circle. Once the car enters the course, it must complete one lap around the right circle to establish the direction of travel, after which it must complete a second lap, which is timed by the judges. After finishing the second lap, the car must move to the left circle to perform two more laps, with the second of those being timed. Once the fourth lap is completed, the car leaves the track.

3. Autocross Event (150pt): The autocross event tests the vehicle's agility and handling on a more challenging closed-loop course. Points are awarded based on the car's lap times and overall performance.

4. Trackdrive Event (250pt): The trackdrive event is an extended track test that evaluates the vehicle's durability and reliability on 10 laps. Points are given for completing each lap, with extra points for completing all laps and faster times.

1.3 The Team Structure

The team "Squadra Corse Driverless" [2] is composed by 45 students divided in 6 different divisions.

The division of "Algorithms for Autonomous Driving" is responsible of the optimization of track mapping, localization of the vehicle, definition of trajectory and reference speed and generally high-level control of the vehicle.

The division of "Vehicle dynamics and Control" is responsible of the State Estimation and all the low-level control of the vehicle, such as Torque Vectoring, Traction Control and Braking Torque split in mechanical and regenerative braking.

The division of “Electrics and Electronics” is responsible of the maintenance of the vehicle from the electrical and electronical point of view and of the development of the Battery Pack and its integration on the car.

The division of “Mechanics and Powertrain” is responsible of the maintenance of the car from the mechanical point of view and of the integration of all the new components in the monocoque. Moreover, it is responsible of the powertrain management.

The division of “Perception” is responsible of the detection of the environment around the car and of developing algorithms to filter and analyze the data coming from sensors such as stereo-camera and Lidar.

The division of “Media and Management” is responsible of the preparation of all the static events and of all the not technical aspects such as sponsorship research, logistic and creation of online contents for the website and social media.

1.4 The Car

The Car is born starting from the chassis of the SC19, prototype developed by Squadra Corse during the season 2018/19. After the born of the team “Squadra Corse Driverless” in 2021, the car has been modified and adapted to driverless task. In the first season of the team, the car renamed “VaLentina” participated to the event of Formula Ata and in this season the new version “Claudia” participated at both the events of Formula Ata and Formula East. A picture of the car with a brief description of some of the main parameters is following.

From a mechanical point of view the chassis is made by a carbon fiber monocoque. The total length of the car is 2874 mm (1525mm the wheelbase), while the track is 1200mm. The total height of the car is 1190 mm. The total mass is 198kg.

From the powertrain point of view the car is driven by 4 in-wheel motors with a maximum power of 80KW and maximum limited torque of 84 Nm. The car can accelerate from 0 to 100 km/h in 2.6 seconds.

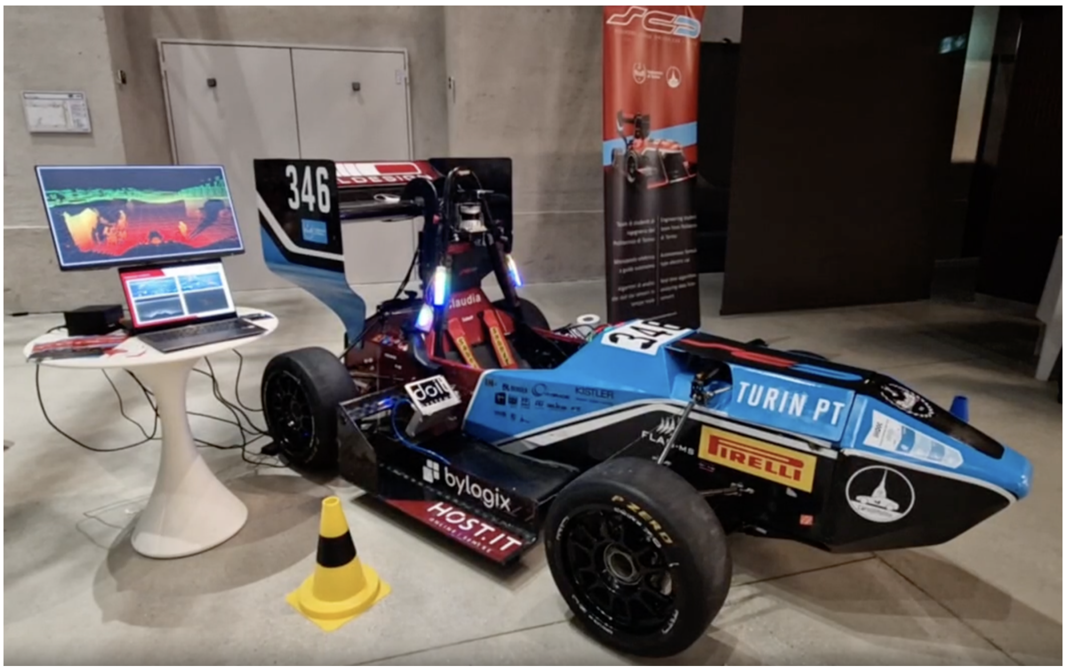

Figure 1.1: The Car ”Claudia” during an exposition Event

1.5 Autonomous System

This section (with references to the Autonomous Design Report of the team for the parteci-pation at Formula Ata [3]) outlines the team’s strategy in creating a comprehensive software architecture for achieving autonomous driving capabilities in the Formula Student Driverless Competition. The approach involves utilizing a LiDAR and two cameras to reliably capture cone positions and colors, thereby defining the track boundaries. These inputs, along with an estimation of vehicle velocity, are processed by a GraphSLAM algorithm to construct a global map of the circuit and determine the current vehicle pose.

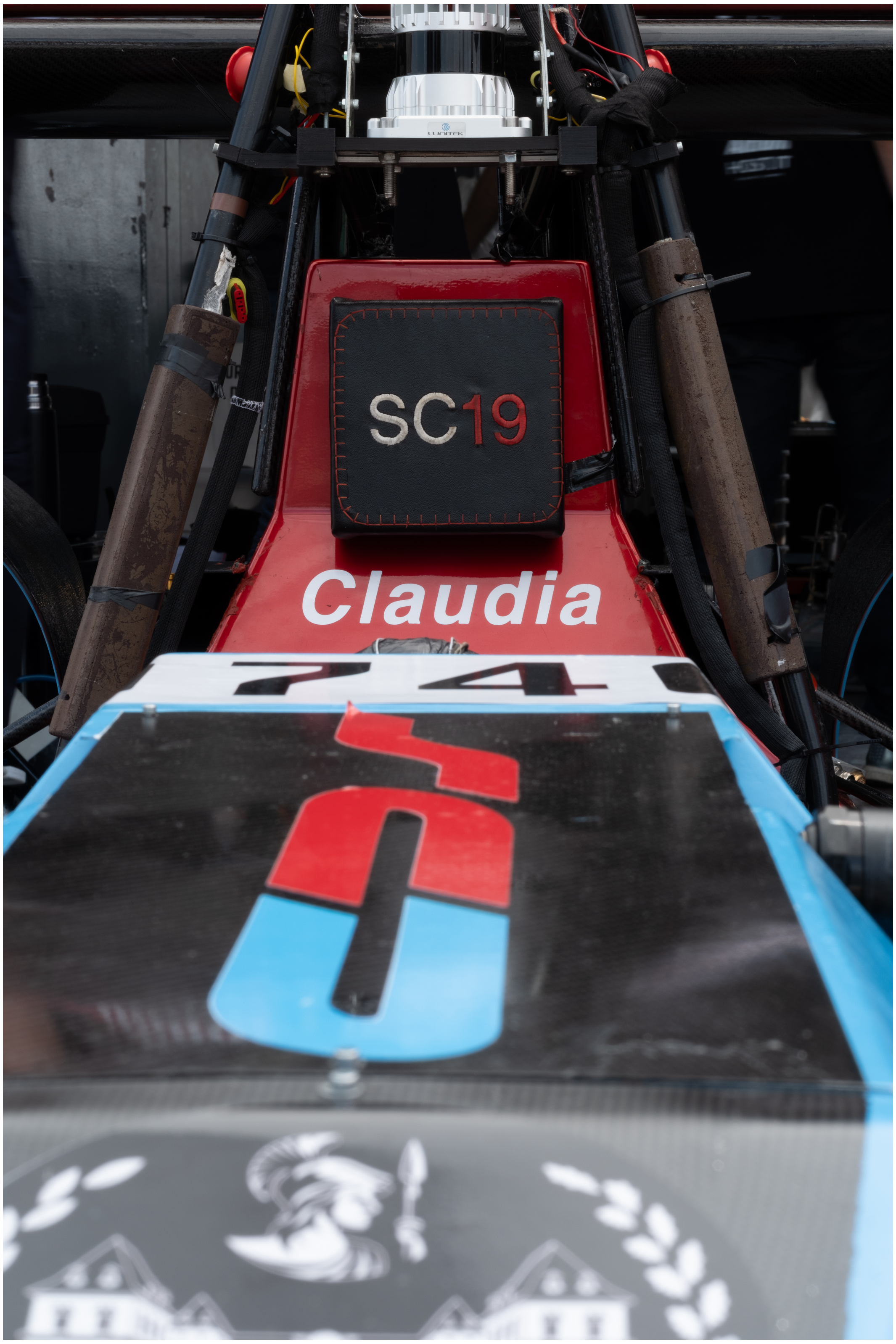

Figure 1.2: The Lidar and the Two Cameras mounted on the car

When the complete map is unavailable, a local planning algorithm is employed to track the middle line. Once the full race track is mapped, a global optimizer calculates the minimum curvature trajectory and devises a velocity profile using a GGV diagram, maximizing the vehicle's performance potential. To adhere to the target path and velocity, lateral Model Predictive Control (MPC) and a Proportional-Integral (PI) speed tracking control are implemented. Additional low-level controllers manage appropriate torque allocation.

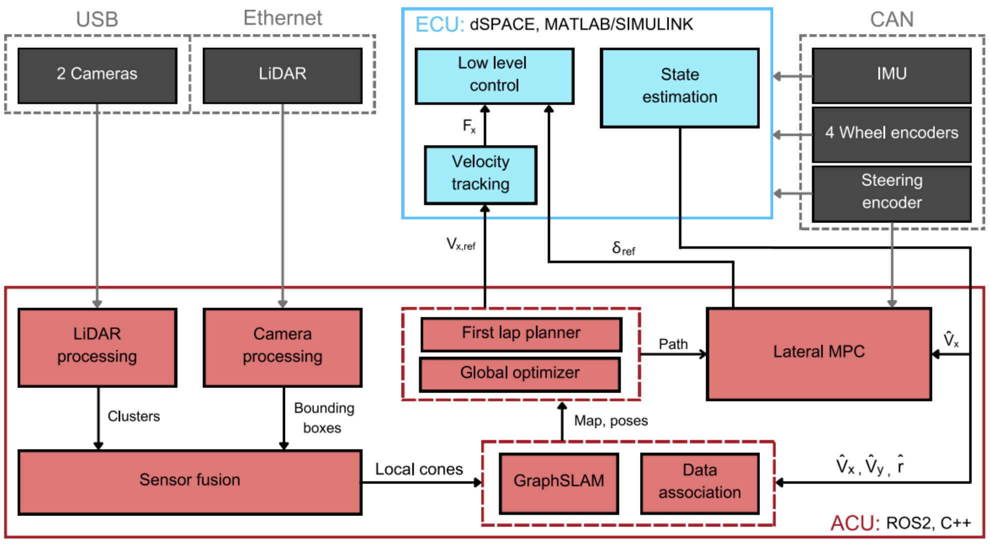

Figure 1.3: Autonomous System Overview

In fig.1.3 an overview of the autonomous system is shown. The system comprises two primary nodes: the Electronic Control Unit (ECU) and the Autonomous Computation Unit (ACU), linked via a CAN interface. The ECU, a dSPACE Microautobox III, operates in real-time with an ARM Cortex A processor and executes MATLAB-Simulink-generated C code. On the other hand, the ACU is a miniITX x86 computer featuring a 13th generation Intel Core i7 CPU and an Nvidia GPU for leveraging multi-threading and CUDA to process the video streams from the cameras. This setup utilizes the ROS2 framework (Humble release) predominantly in C++.

Each ROS node operates within its dedicated docker container, which undergoes nightly builds as part of the Continuous Integration pipeline. This workflow streamlines code deployment, reducing development time significantly. The ACU's primary task involves collecting and processing raw data from a LiDAR and two cameras to extract cone positions and colors within the environment. This information, along with the estimated vehicle states, is then relayed to the SLAM (Simultaneous Localization and Mapping) module, generating a global map of the environment utilized by the path planner to calculate the target trajectory.

High-level controllers interpret the trajectory and translate it into steering and torque setpoints. Subsequently, low-level controllers deployed on the ECU fine-tune these setpoints, ensuring optimal performance and delivery to the actuators.

The primary goal of the perception pipeline is to identify cones within the environment and derive estimations regarding their positions and colors. The SCD23 vehicle is equipped with an Ouster OS-1 LiDAR featuring 64 channels operating at a frequency of 10Hz. Additionally, it incorporates two Alvium 1800 U-507 RGB cameras mounted atop the vehicle's main hoop. Both sensors are slightly angled, with the cameras oriented in a forward-facing setup, each at different angles, collectively providing a Field of View (FoV) spanning 120°.

The sensor fusion algorithm integrates data from both the cameras and LiDAR systems to generate comprehensive information regarding the positions of cones on the track.

For associating LiDAR centroids with the center of the bounding boxes obtained from the object detector, a proprietary point-to-point association algorithm was developed. This algorithm utilizes the roto-translation matrix of the sensors' frames and the intrinsic parameters of the cameras to perform the necessary 3D to 2D projection for data association. To enhance reliability, the projected points undergo filtering using a dynamic distance metric, which relies on the vertical deviation of the image from the horizon.

To ensure the robustness of the approach, an Iterative Closest Point (ICP) algorithm was initially incorporated. However, after thorough testing, it was found that it did not yield any noticeable improvements to the existing sensor fusion pipeline. Therefore, it was omitted from the final release of the system.

1.6 Autonomous Driving: SLAM

The term SLAM refers to the set of algorithms designed to simultaneously localize the robot (i.e. car) on the racetrack and create a map of the environment, characterized by fixed features (i.e. cones) called landmarks.

Graph-based SLAM involves constructing a graph where nodes symbolize both the robot's poses and the landmarks, and edges depict the connections between these nodes. The primary objective is to determine an optimal node configuration that effectively fulfills all the constraints within this graph.

The construction of the graph, called front-end, is performed as follows: for each odometry acquisition $u_{t,t-1}$, a new node is added to the graph corresponding to the pose x_{p_t}, and an edge is inserted between $x_{p_{t-1}}$ and x_t corresponding to $u_{t,t-1}$. For every landmark measurement $z_{i,j}$, a data association algorithm is executed. If a correspondence is detected between the measurement and an existing node, a constraint is established between the two nodes, corresponding to the measurement $z_{i,j}$. If no correspondence is found, a new node is added to the graph corresponding to a new landmark x_{l_i}.

Data Association plays a crucial role in the SLAM problem as the optimization of pose heavily relies on recognizing multiple observations of the same landmark. Given that every cone has the same appearance, using a feature extraction algorithm to identify repeated landmarks is not a feasible option. The remaining choice is between Bayesian or non-Bayesian approaches for data association. Among the Bayesian methods, Multiple Hypothesis Tracking (MHT) is a commonly used example. However, implementing MHT can be challenging and computationally expensive. On the other hand, the k-nearest neighbors (kNN) approach is more straightforward but may encounter issues if the landmarks are not well separated and is susceptible to noise.

The best results have been obtained using a Joint Compatibility Branch and Bound (JCBB) algorithm [4], which assigns every new data point to one of the points already present in a recursive manner. Compatibility pruning is achieved by considering the Mahalanobis distance both from the new point to the assigned point (referred to as individual compatibility) and between all assigned points as a whole (known as joint compatibility). A modified version of the JCBB algorithm has been chosen and implemented to reduce the complexity of the joint compatibility test from $O(n^2)$ to $O(1)$.

The trajectory planning module aims to identify the best route within the track's boundaries that adheres to the vehicle dynamics constraints. Leveraging SLAM-derived position and color data of landmarks, a multi-step approach is employed. Initially, the spatial domain is discretized using Delaunay Triangulation, enabling easy definition of the middle line by center points of edges connecting different-colored vertices. From the car's position, an offline-generated tree of potential paths is loaded. Subsequently, each branch of the tree is associated with a cost function, and the most suitable branch for tracking the middle line is chosen as the optimal trajectory.

This process exhaustively explores the trajectory space while ensuring real-time computational feasibility and deterministic behavior, mitigating issues linked to randomness. This significantly reduces computational load compared to the previous solution, which relied on the online rapidly exploring random tree (RRT) method.

Upon loop closure detection by SLAM, critical information defining track boundaries is provided, allowing subsequent application of optimization techniques to generate the desired racing line. The implementation is grounded in minimizing curvature.

To achieve optimal performance while ensuring stability, a speed profile is crafted using characteristics from the global optimal line and tire grip modeling based on GGV diagrams.

In order to track the reference trajectory as fast as possible, a motion controller consisting of a high- and a low-level controller has been developed. At high-level a Model Predictive Controller (MPC) governs the lateral vehicle dynamics while a PI controller is used for speed tracking. At low-level a traction control (TC) and torque vectoring control (TVC) have been implemented to increase stability.

1.7 Thesis Outline

The Autonomous System of the car has been briefly described in previous sections. A quick introduction to SLAM Algorithm has been done and more details are available in the next chapter.

The second chapter is fully dedicated to the explanation of the SLAM algorithm and the introduction of the actual State of the Art related to the State Estimation. All the aims of the activity are presented in this chapter with particular focus on the implementable solutions on the car. Different methods different from State Estimation are also presented.

The third chapter is related to the Longitudinal Speed Estimation. Particular focus will be given to the car architecture, team targets and actual state of the art. Then, the adopted strategy will be described with particulat focus on results obtained on simulation environment and on track.

The next chapters are dedicated to the estimation of the sideslip angle of the car. In Chapter 4. an introduction to the general adopted strategy is followed by the explanation of the Kalman Filter algorithm. Then a kinematic model is introduced and all the steps of the Kalman filter are explained. Some considerations about the results obtained will be done.

In Chapter 5. a Dynamic Kalman Filter algorithm is explained. Firstly, some theory concepts about vehicle dynamics will be introduced, followed by an introduction to tire behavior (lateral dynamics). Then the choosen dynamic model is described with focus on all the steps of the Kalman Filter. The results obtained are compared with the ones of the kinematic model.

In Chapter 6.the implementation of the dynamic filter for real application is discussed and an innovative method to combine the results of the two filters is introduced and discussed.

In Chapter 7. the results obtained for real application are discussed and compared with the ones obtained with just the use of the dynamic filter.

In the Chapter 8 it is introduced the algorithm to estimate the vehicle position and the track mapping through the Estimation. Firstly, an analysis of the data obtained with the Ground Speed Sensors is done, followed by the comparison of the results obtained in real time application and by estimation.

Chapter 2

State Estimation: Aims and Targets

2.1 Aims and Introduction to SLAM

In driverless application, due to the absence of the driver, the knowledge of the correct position of the vehicle on the track is fundamental. GPS does not represent a competitive solution due to its accuracy of few meters. The width of track (minimum 3m) is not compatible with the accuracy of GPS. Moreover, while in acceleration and skidpad events the track is well known in advance, in autocross and trackdrive events the layout is completely unknown and the correct position of the vehicle is more important because it is necessary to map the track with all its characteristics (length of straights, radius of curvature, presence of hairpins etc.). For this reason, the localization and the mapping should not be computed directly by position measurements but should be performed by the integrations of other measurements and sensors in another way.

The solution chosen by the team is SLAM (Simultaneous Localization and Mapping) algorithm. SLAM [5] is a method used for autonomous vehicles that allows them to simultaneously build a map and locate the vehicle within that map. SLAM algorithms enable the vehicle to map unknown environments.

In general, there are two categories of technological components utilized in creating SLAM. The first category involves sensor signal processing, encompassing front-end processing, which is largely dependent on the sensors being used. The second category pertains to pose graph optimization, including back-end processing, which operates independently of the sensor type.

SLAM LIDAR [6][7], which stands for Light Detection and Ranging, is a technology primarily reliant on laser or distance sensors. In comparison to cameras, Time-of-Flight (ToF) sensors, and other sensing methods, lasers offer notably higher precision and find extensive use in applications featuring high-speed moving vehicles, including autonomous cars and drones. Laser sensors typically generate output data in the form of 2D (x, y) or 3D (x, y, z) point clouds. These point clouds from laser sensors deliver exceptionally precise distance measurements and are exceptionally effective in constructing maps for SLAM (Simultaneous Localization and Mapping). Moreover, the process of matching these point clouds often requires substantial computational power, making process optimization crucial for improving speed. Considering these challenges, autonomous vehicle localization often involves integrating additional measurement data sources, such as odometry, Global Navigation Satellite System (GNSS), and Inertial Measurement Unit (IMU) data.

For this reason, one of the main targets to achieve good results in SLAM is the correct acknowledgement of the velocity of the vehicle (longitudinal and lateral) as well as the yaw rate of the vehicle.

2.2 Ground Speed Sensor and other solutions

As stated in previous subsection, the correct and instantaneous knowledge of longitudinal and lateral velocity of the vehicle is fundamental. Unfortunately, the only way to directly measure these quantities is the use of a Ground Speed Sensor (GSS). This solution, due to the very high cost of this kind of instrument, is not suitable with the budget of a FSAE team like "Squadra Corse Driverless Polito". Mainly for this reason, other solutions have been exploited by the team. The two principal solutions adopted are State Estimation (explained in this work) and the fabrication of a Ground Speed Sensor custome. Both the solutions have been carried on for all the season and their fusion is one of the next steps.

The custome Ground Speed Sensor's basic idea is related to the functioning of the commercial instrument. A GSS is basically an optical sensor. The main idea consists of the use of a camera that takes photos at very high frequency (hundreds of Hertz), then the Fourier transform is applied to all the pictures. After a filtering in the frequency domain, the longitudinal and lateral displacement by two pictures is calculated. Knowing the time at which each photo has been taken it is possible to know the components of the instantaneous velocities. Very good results have been achieved by the team with this kind of solution.

State Estimation consists, instead, in the development of algorithms, mathematical and physical models and other tools in order to estimate the longitudinal and lateral components of the velocity starting by other measurements coming from other sensors such as IMU, steering wheel and rotational wheel speed sensors. An introduction that describes all the possible solutions and the actual state of the art is following, while all the work done will be explained in next chapters.

2.3 Actual State of the Art

As stated in previous subsection, a direct measurement of vehicle velocity and vehicle sideslip angle is available on the market, but it is not suited to most of the application (even for passenger and luxury cars) due to the high cost of the required sensors. By the way, alternative methods to estimate or measure indirectly these quantities have been developed in the recent past years and most of them are used in common passenger cars.

Two distinct approaches can be employed for this purpose: analytical techniques, which are model-based methods (such as observer or filter techniques), and empirical techniques that rely on artificial intelligence.

Model-based virtual sensing techniques utilize a mathematical model of the system to estimate missing values based on the evolution of other measured parameters. One example of this approach is the Kalman filter, which is employed to estimate the state of linear time-invariant systems. However, these techniques require the availability of a mathematical model of the system to function effectively. There are two primary issues associated with this method. Firstly, the model needs to be accurately calibrated, with correct parameter values. Secondly, the model must encompass the entire range of operating conditions to avoid unmodeled dynamics. Incorporating an analytical description into the model that can adequately represent a system's behavior across all possible operating conditions is a challenging task, particularly for complex systems like vehicles.

Another approach is to utilize empirical techniques, which operate without relying on a model. In this case, the system is treated as a black box and is not described by mathematical equations. In simpler terms, an artificial neural network is trained to predict the output corresponding to a given set of inputs, relying on empirical observations rather than a mathematical system description. This method is gaining popularity because model-based virtual sensing can have limited accuracy due to the critical issues mentioned earlier. Neural networks are mathematical representations inspired by the functioning of our brains. They consist of nodes (neurons) organized into multiple layers that activate in response to specific input sets, each performing different operations on the input signals. The number of nodes depends on the system's complexity and the desired level of accuracy.
Given a set of inputs, there are numerous possible combinations of nodes to produce the output. However, the issue of which nodes should be activated is unequivocally resolved at the input layer level. When a set of inputs is provided, the neural network automatically determines which nodes should be activated. The challenge lies in teaching the neural network how to do this.

This teaching process occurs during supervised training. In this phase, a set of inputs is provided to the network, and the corresponding correct outputs (y) are known. The network evaluates all possible combinations of nodes to generate various estimations ($\hat{y}$) of the output. The correct combination of neurons minimizes the estimation error ($e = y - \hat{y}$). Once this occurs, the neural network learns which combination of nodes should be activated when that particular input set is provided. This process requires a sufficiently extensive dataset of experimental values obtained through empirical procedures on the actual system. However, a limitation of this approach is that a substantial amount of experimental data is required to train neural networks, and this data is specific to the particular vehicle model from which it was collected. Consequently, implementing artificial neural network techniques demands a significant upfront effort.

2.3.1 Empirical Models State of the Art

In most of common passenger cars, the implementation of neural networks is slightly increasing over the years. Estimation through empirical models show greats advantages in market application. Due to the high number of vehicles sold for the same model, in fact, it is necessary just to calibrate the model on one single vehicle, exploiting almost all the most common driving conditions. Moreover, the absence of knowledge or the high variability of some parameters (such as the total mass depending on number of passengers and luggage or the stiffness of tires depending on driving condition) lead to the preferring of empirical methods respect to mathematical ones. At the actual state of the art a lot of categories of Neural Network to estimate the sideslip angle are present.

Kato et al. [8] apply a method that combines the NN output with the ones of kinematic and dynamic models. Others paper, such as the ones proposed by Sasaki et al. [9]

propose the use of past input and a time delay NN. A lot of other methods are present in literature. Moreover, since the input of the NN are data coming from sensors other methods are exploited to improve robustness and vehicle state identification. Bonfitto et al. [10], for example, propose a NN that identifies the road condition (ice, wet and dry), while Martino et al. [11] propose the use of Principal Component Analysis for Data pre-processing.

In [10]the sequence of steps occurs within a complex system involving speed and road condition estimation. It starts with a set of measured values feeding into three neural networks, each specialized in estimating vehicle speed under specific road conditions (dry, wet, or icy). The road condition identification block, equipped with a neural network, discerns the prevailing road conditions. This output is then used by a selector to determine which of the three estimated vehicle speeds to utilize.

The estimated speed feeds into calculating the longitudinal slip of the four wheels. Once established, this dataset enters the sideslip angle estimation block. Similar to the speed estimation, three distinct neural networks are involved, each trained to estimate the sideslip angle under specific road conditions. The road condition identification block again comes into play to decide which of the three estimations is pertinent in the current road scenario.

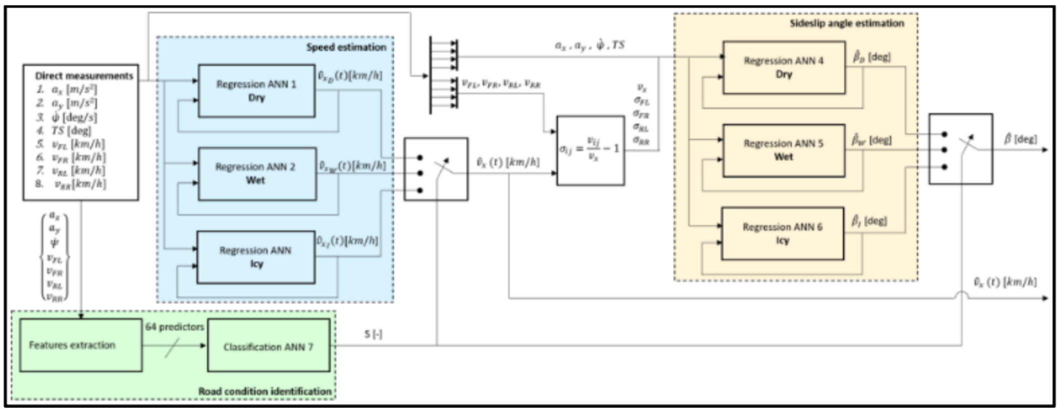

Figure 2.1: Scheme for Implementation of Neural Network for identification of road condition [10]

In estimating a variable, another method employed is the usage of rule-based techniques, which are model-less approaches relying solely on the knowledge and expertise of the designer. These methods utilize the designer's understanding of the vehicle's behavior in specific conditions, described through linguistic statements rather than mathematical equations. Fuzzy logic, a prevalent method in this category, emphasizes the 'non-rigorous' nature of this approach. It results in an approximate representation of the problem, rooted in assumptions made by the designer. Rule-based methodologies find widespread use in various industrial applications due to their simplicity of implementation, provided a robust understanding of the system's physics.

An illustrative instance of estimating vehicle speed through a fuzzy logic approach involves computing an average of the velocities of the four wheels is following. However, the inclusion of wheels in this average computation depends on the reliability of their measurements. Reliability coefficients, denoted as ΔS values, are established by comparing the current wheel velocity with that from the previous time instance. If the difference in velocity for one wheel significantly exceeds the differences observed in the other wheels, it signals potential unreliability in the measurement for that particular wheel. Consequently, such unreliable measurements are excluded from the averaging process. Thus, at each time instance, a reliability index is assigned to individual wheel speed measurements to determine their inclusion in the averaging procedure for estimating the vehicle's speed.

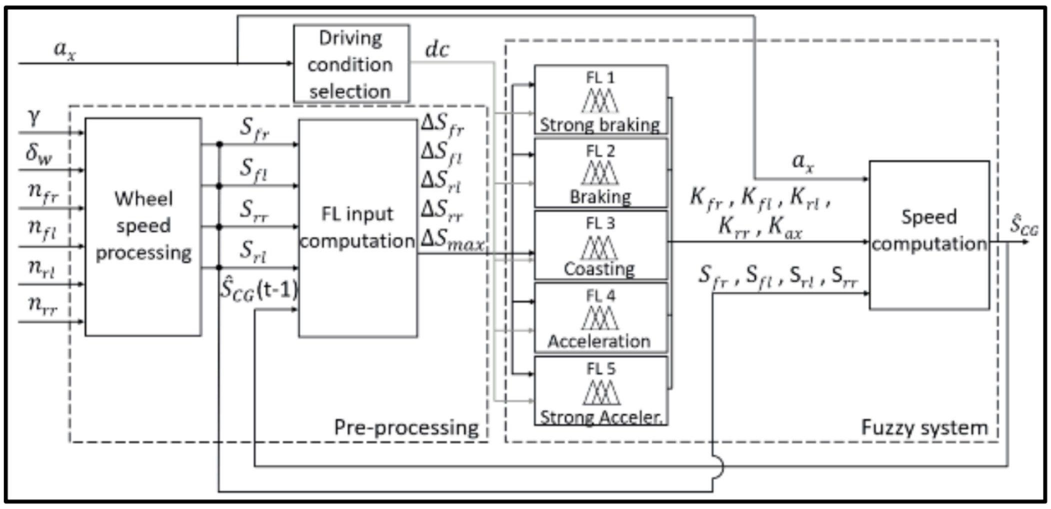

Figure 2.2: Implementation of Fuzzy Logic for vehicle Speed estimation[10]

2.3.2 Physical Model State of the Art

Most of all the proposed methods are applied in passenger cars. Anyway, the need of a high number of data covering all possible driving condition does not match with the requirements of the team. Moreover, most of the estimators of sideslip angle in passenger cars have different aims (stability and help to drivers) than the ones required by the team and so the use of empirical methods could not be the best solution at the moment for the team. For this and other reasons, the adopted strategy consists of the use of mathematical and physical models, since most of the problems present in passenger cars are not present in the vehicle of a FSAE team (the mass is constant, the file .tir of the tires are well known etc.).

Quoting [12],numerous observers have been developed to this aim, yet sideslip angle estimation is still an open issue in the automotive field.In the literature, three modeling categories can be identified: kinematic models, dynamic models, and combined models. Most of the proposed methods to estimate sideslip angle need only signals that are easily measurable within the integrated set of sensors already available in a standard passenger car.

The first category relies on kinematic relationships involving yaw rate, lateral and longitudinal velocities, and their derivatives. It does not consider any vehicle or tire parameters. However, kinematic-based estimators tend to become unobservable when the yaw rate approaches zero, often resulting in noisier estimations. Nonetheless, kinematic models excel in transient maneuvers and prove effective in the nonlinear region of the tire.

An example of this is [13] where a straightforward yet effective logic is employed to address the unobservability issue and prevent potential sideslip angle drifts on straight roads due to inevitable sensor offsets in yaw rate and lateral acceleration. Selmanaj et al. [13] corrected the approach previously proposed by adopting a heuristically calculated term. This heuristic function is assessed through bivariate Gaussian distributions and a set of three signals—steering angle, yaw rate, and sideslip rate—along with their derivatives.

Estimators solely relying on kinematic models are infrequently encountered due to their limitations. Instead, the prevalent approach involves either a dynamic model or a combination of kinematic and dynamic models.

The second category is extensively found in the literature, based on the vehicle's equilibrium equations often depicted via a single-track model. Instances employing a four-wheel configuration vehicle model are also documented. Frequently, an Extended Kalman filter is employed, with the Unscented version used in highly nonlinear models. Dynamic methodologies are notably influenced by the tire model integrated within the estimator. Some studies opt for a linear tire model, while others make use of Pacejka's Magic Formula.

The third category features a mixed approach, employing a well-thought combination of kinematic and dynamic modeling.
All the work done is freely inspired by other papers and models proposed by some authors such as Lenzo [12], Savaresi [14] and Rajamani [15]. All of them propose the use of kinematic and dynamic models and how implement them in passenger cars. This work tries to find the best approach for Formula Student Driverless application, starting from all the proposed solution. This work is properly focused on the development of a kinematic and dynamic model based on the available sensors and on the fusion of the two models. Particular attention will be given to tire behavior modelling, different dynamics model, the use of the Kalman Filter and its formulation, in the application on the vehicle and on results. All the work is based on a validation on simulation through the software Vi-Grade [16] and then on a validation through some data collected on track. In conclusion an analysis of the results and how they influence the correct localization of the vehicle on track will follow.

Chapter 3

Longitudinal Speed Estimation

3.1 Targets

As stated in previous chapters, two main goals must be achieved in the State Estimation work: the estimation of the longitudinal speed of the vehicle and the lateral speed (knowing the longitudinal velocity knowing the sideslip angle is sufficient). All the proposed work is mainly focused on the estimation of these two quantities. In this chapter all the developement of the longitudinal velocity estimation will be described with particular attention to results obtained in simulation and on track.

In this section, all the work related to the longitudinal velocity estimation will be explained, starting from theory concepts, required sensors and methodology.

3.2 Theory concepts, actual state of the art and vehicle structure

Tires are one of the most important parts of a vehicle. They represent the only way to exchange forces with the ground. In traction and braking conditions, positive and negative forces are exchanged (at the moment lateral dynamics is neglected). As it is well known, longitudinal forces are exchanged when a longitudinal slip is present. Different definitions of slip are well used in the literature, some of them differ between traction and braking conditions, but most of them lead to the same results. One of the most common definitions of slip is given by the following formula:

$$s = \frac{\omega - \omega_0}{\omega_0} \tag{3.1}$$

where ω is the rotational speed of the wheel and ω_0 is the rotational speed that the wheel should have in order to not exchange longitudinal forces. Similar definitions have ω at the denominator, or in some definitions, ω and ω_0 are substituted by the longitudinal speed of the contact patch and the longitudinal speed of the center of the wheel. The way in which forces are exchanged is represented in the following qualitative graph.

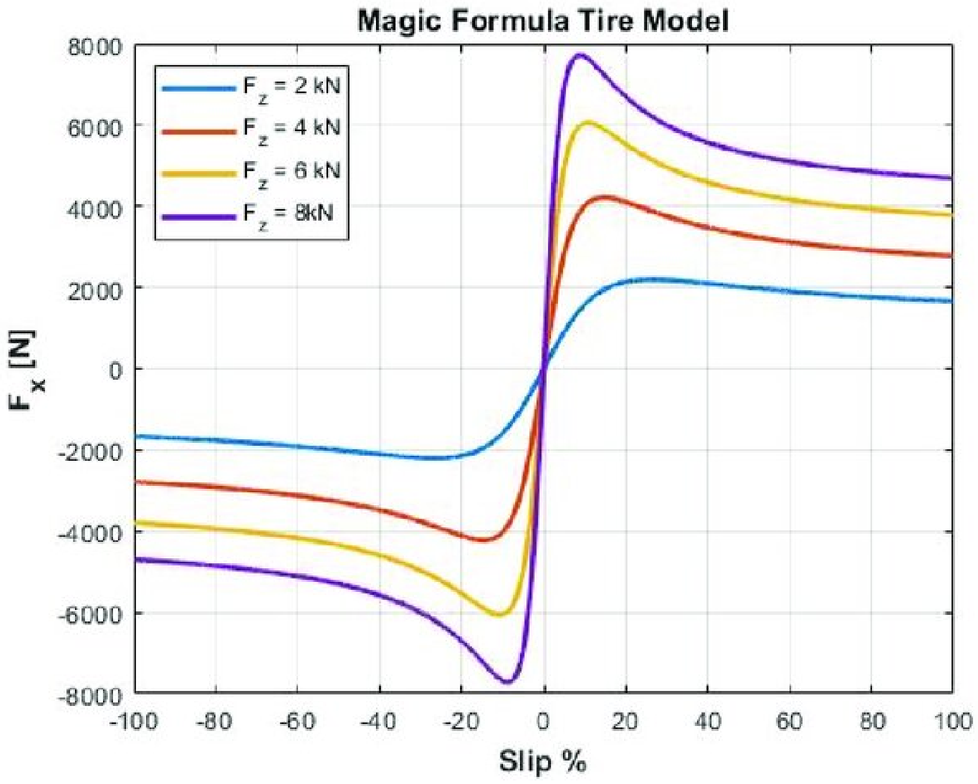

Figure 3.1: Generic Force Slip Diagram

Credits of graph to [17]

For this reason, it is intuitive to say that for a race car, where traction and braking request could be very high in absolute value, the velocity of the wheel and the velocity of the vehicle do not perfectly match and so the velocity of the vehicle cannot be estimated, using rotational speed of tires only, but an algorithm or another model is necessary.

Moreover, in all driving conditions, but particularly in cornering, the motion of a vehicle is not just translatory but it has also a rotatory component that leads to the fact that each part of the vehicle has its own velocity and that even the speed of the center of the wheels will be different to the one of the center of gravity of the vehicle accordingly to the driving conditions.

Most of common passenger cars have only two driving wheels (FWD front wheels and RWD rear wheels). For this reason, on traction mode, only driving wheels have slip and so the velocity of the center of the wheels is easily computable measuring the rotational speed of non-driving wheels. In braking condition, the situation is slightly different since braking torque is generally applied on all the wheels.

At the actual state of the art, different ways to estimate longitudinal speed are present on common passenger cars. If high accuracy is not needed, low-cost car can simply estimate the velocity of the vehicle starting from rotational speed of non-driving wheels or just from the rotational speed of the engine (if the gear is known). Other strategies can rely on the rotational speed of all four wheels, excluding wheels that have an output much different from the others. A solution like this, due to its low accuracy, does not represent the best solution for FSAE application.

In modern passenger cars, and where higher accuracy is required, more sophisticated systems are present to estimate the longitudinal velocity of the vehicle. Modern vehicle, in fact, require the knowledge of this state, not only just to communicate the value to the driver, but also to activate and manage controls such as Traction Control, Abs, Cruise Control and Lane Keeping.

Usually, the estimation is given by a Neural Network, that starting from the output of the sensors (RPM, steering wheel, pedal travel, IMU) can recognize road condition and speed of the vehicle [10]. This solution could be very powerful, but unluckily it requires a lot of tests and training and so it does not represent the best solution for the team "Squadra Corse Driverless Polito", but a solution that considers the needs and the available sensors of the car is needed.

3.3 Adopted strategy

As stated in the introduction, "Claudia" is driven by 4 in-wheel electric motors, i.e. each motor rotational speed is strictly related to the speed of that wheel and vice versa. Sensors that measure the rotational speed of the motors are available, so sensors that directly measure the rotational speed of the wheel are not needed.

The motor rotational speed is acquired from the RPM sensor (for each motor) and then is converted into the linear speed of each wheel (the wheel radius is assumed to be constant). Then the data coming from the IMU sensor are acquired and filtered. The linear speed of each wheel is then converted respect to the centre of gravity of the vehicle (for longitudinal velocity estimation we assume the sideslip angle equal to 0). The formula for each tire is reported.

$$V_{\text{cog}} = V_{\text{fl}} + \dot{\Psi} \cdot \frac{T_f}{2} \tag{3.2}$$

$$V_{\text{cog}} = V_{\text{fr}} - \dot{\Psi} \cdot \frac{T_f}{2} \tag{3.3}$$

$$V_{\text{cog}} = V_{\text{rl}} + \dot{\Psi} \cdot \frac{T_r}{2} \tag{3.4}$$

$$V_{\text{cog}} = V_{\text{rr}} - \dot{\Psi} \cdot \frac{T_r}{2} \tag{3.5}$$

In this formula, V_{ii} represents the speed of the ii tire, while $\dot{\Psi}$ represents the yaw rate measured by the IMU sensor, and T_f and T_r are respectively the front and rear track (distance between the left and rear wheel).

Then the speed is estimated according to a weighted average of the four linear velocities and the past value of velocity plus the longitudinal acceleration multiplied by the time interval.

$$V_{\mathrm{cog}} = \frac{V_{\mathrm{fl}} \cdot K_{\mathrm{fl}} + V_{\mathrm{fr}} \cdot K_{\mathrm{fr}} + V_{\mathrm{rl}} \cdot K_{\mathrm{rl}} + V_{\mathrm{rr}} \cdot K_{\mathrm{rr}} + (V_{\mathrm{last}} + a_x \cdot Ts) \cdot K_l}{K_{\mathrm{fl}} + K_{\mathrm{fr}} + K_{\mathrm{rl}} + K_{\mathrm{rr}} \cdot K_l} \tag{3.6}$$

In the previous formula, V_{ii} does not represent the linear velocity of the ii tire, but the velocity of the center of gravity of the vehicle calculated using the linear velocity of the ii tire.

The five coefficients are not constant, but they vary accordingly to a fuzzy logic dependent on driving conditions (normal driving, braking, strong braking, acceleration, strong acceleration, and cornering).

3.4 Fuzzy Logic

As stated in the previous subsection, the weight of the five terms is not constant but depends on the driving condition. The selection of the driving condition depends on the output of IMU sensors, after filtering. Particularly, all driving conditions are classified into the following categories:

- Strong acceleration (if the longitudinal acceleration sensor output ax is >0.8 g)
- Acceleration (if the longitudinal acceleration sensor output ax is <0.8 g and >0.3 g)
- Cornering (if the longitudinal acceleration sensor output ax is >0.3 g and < -0.3 g)
- Braking (if the longitudinal acceleration sensor output ax is < -0.3 g and > -0.8 g)
- Strong Braking (if the longitudinal acceleration sensor output ax is < -0.8 g)

If the output is not reliable, or other cases are exploited, the driving condition zero is enhanced.

In each of the previous conditions, the rules to determine the weight of each coefficient are similar. Firstly, the output of each RPM encoder is taken and converted into the linear speed of the wheel and then converted into the velocity of the center of gravity of the vehicle (as explained in the previous subsection). This value is compared to the previous estimated value of the velocity of the car (0 if it is the first iteration). If a high difference is detected, or a difference not compatible with accelerometer output (for example, a speed lower than the previous while a positive acceleration is measured), the output of the encoder is declared unreliable, and the weight coefficient for that wheel is forced to 0. Otherwise, if the output is declared reliable, the weight coefficient for that wheel is forced to a value dependent on the driving conditions.

The general rules of fuzzy logic have been explained. More information about the determination of the value of each coefficient in all driving conditions are out of the scope of this work.

3.5 Required Sensors

The purpose of the velocity estimation is, as already stated, the knowledge of the correct value of the vehicle's speed, despite the absence of a direct measurement of the quantity.

For this reason, other sensors have been deployed to obtain this estimation. The required sensors for this approach are introduced:

- **IMU (Inertial Measurement Unit)**: This sensor must be placed as close as possible to the center of gravity of the vehicle. The output of this sensor includes the three components of acceleration (longitudinal, lateral, and vertical) and the yaw rate of the vehicle. The vertical component is not used in the adopted strategy (road inclination and roll motion effects are neglected). The filtering (and its effect) of this sensor will be explained in the next sections.

- **Steering Wheel Position Sensor**: This sensor measures the angular position of the steering wheel. The angular position of the wheels cannot be measured, but it is estimated accordingly to the angular position of the steering. This quantity is necessary to calculate the longitudinal components of front wheels.

- **RPM Motor Sensor**: This sensor is used to measure the angular velocity of the four in-wheel motors. Obviously, one sensor for each motor is present. The conversion of this quantity into linear velocity and its reliability have been discussed in previous subsections.

3.6 Validation and Results

The proposed model and the adopted strategy have been discussed. For validation the model has been validated on some simulations on the software Vi-Grade [16] and then tested on track (validation through a Kistler Ground Speed Sensor [18]).

As far as it concerns simulation, test strictly related to FSAE applications have been done (skidpad and autocross). Moreover, to test the reliability and the response of the model, for skidpad two different tests have been done. In the first test, the vehicle started with a zero-velocity, while in the second it started with a higher velocity. The first test is more representative of real application, while the second one represents the ability of the model to reach as close as possible the correct value, even if the starting condition (high real speed and zero estimated speed) were different.

The results are now shown and discussed. For each test, rms and mean absolute value errors will be calculated. Moreover, a percentage of goodness of fit will be evaluated accordingly to the following formula:

$$\text{Fit } (\%) = 100 \cdot 1 - \frac{||V_{\text{est}} - V_{\text{real}}||}{||V_{\text{real}} - \text{mean}(V_{\text{real}})||} \quad (3.7)$$

The following graph represents the behavior on a skidpad lap, with starting velocity equal to zero.

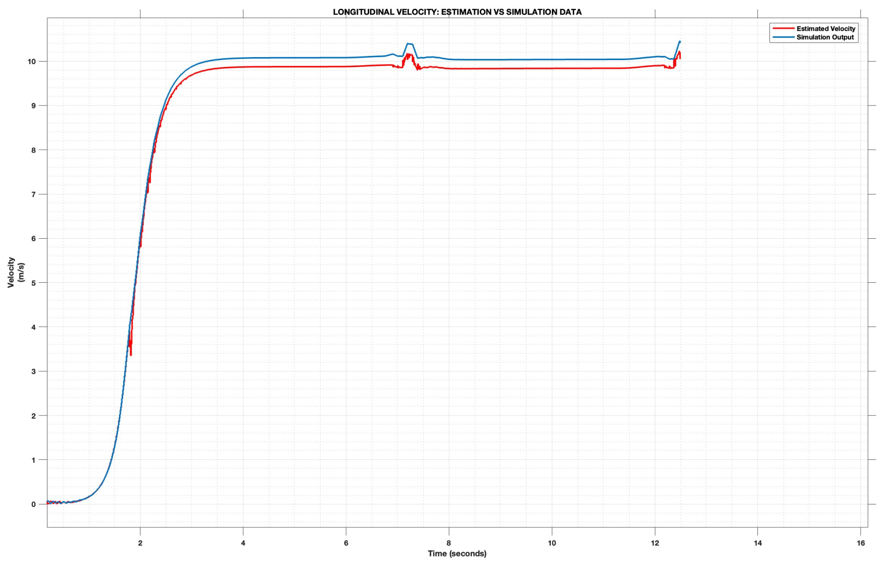

Figure 3.2: Longitudinal Speed Estimation: Vi-Grade Test1

Table 3.1: Simulation Results: Longitudinal Velocity error: Test 1

Error Type	Value in m/s	Value in km/h
Root Mean Square (RMS) Error	0.1912	0.6884
Mean Error	0.1731	0.6232
Mean Absolute Error	0.1752	0.6309

$$\text{Goodness of Fit} = 94.33\% \tag{3.8}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Vx_{est} - \text{mean}(Vx_{est}))}\right) \tag{3.9}$$

The following graph represents the behavior on a skidpad lap, with starting velocity not equal to zero.

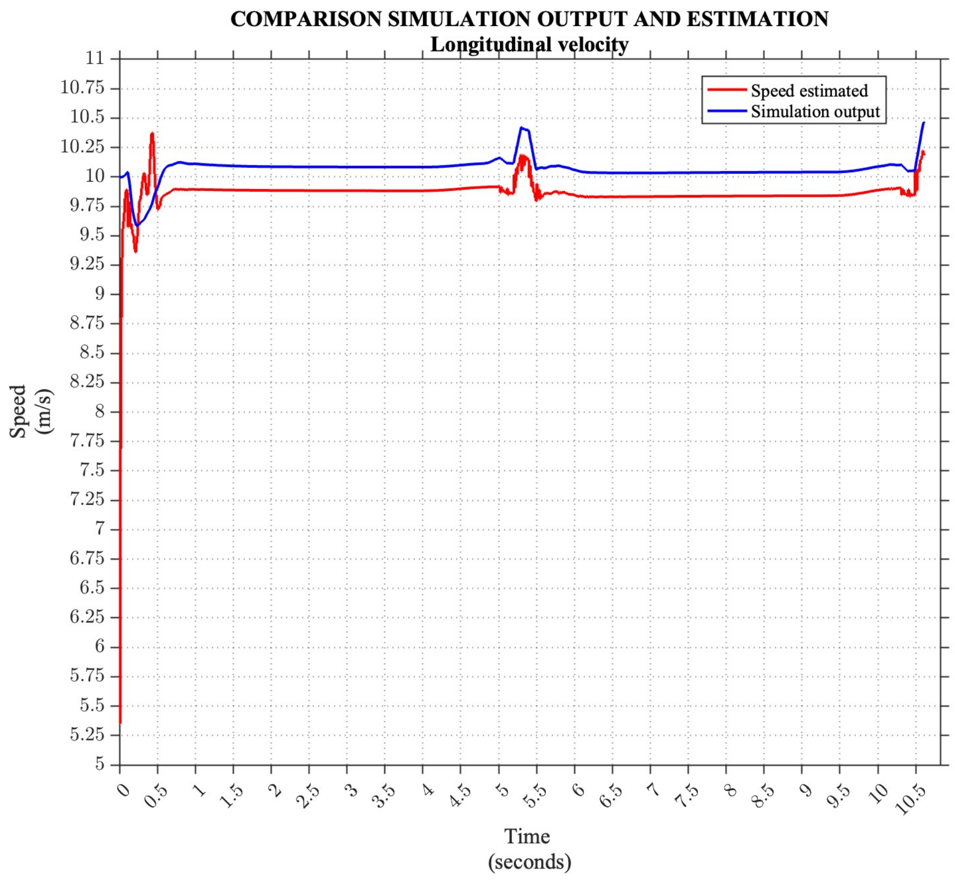

Figure 3.3: Longitudinal Speed Estimation: Vi-Grade Test2

Table 3.2: Simulation Results: Longitudinal Velocity error: Test 2

Error Type	Value in m/s	Value in km/h
Root Mean Square (RMS) Error	0.2686	0.9669
Mean Error	0.2034	0.7322
Mean Absolute Error	0.2168	0.7805

$$\text{Goodness of Fit} = 93.27\% \tag{3.10}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Vx_{est} - \text{mean}(Vx_{est}))}\right) \tag{3.11}$$

Even if out of the team's actual application a test with very high velocity and longer duration has been done on a complete autocross event.

The following graph represents the behavior on this event.

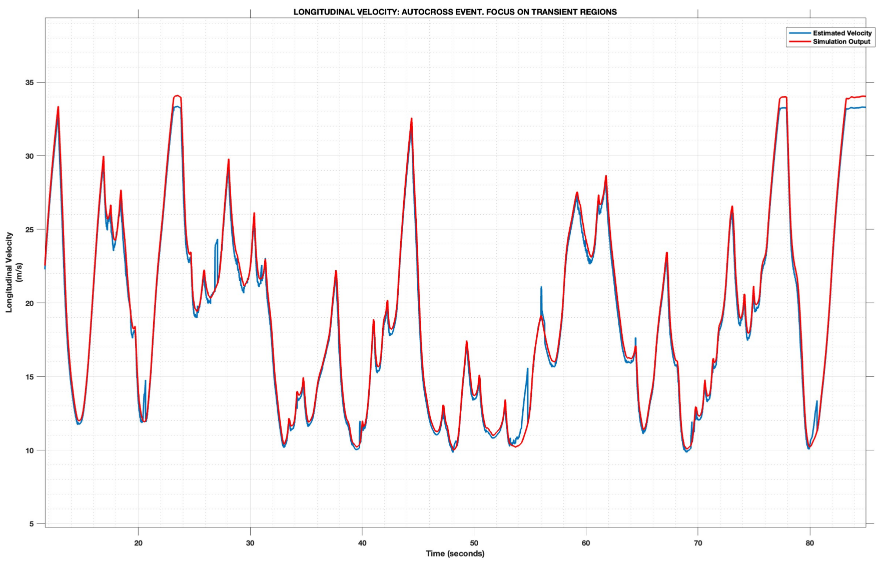

Figure 3.4: Longitudinal Speed Estimation: Vi-Grade Autocross Event

Table 3.3: Simulation Results: Longitudinal Velocity error: Autocross Event

Error Type	Value in m/s	Value in km/h
Root Mean Square (RMS) Error	0.6042	2.1749
Mean Error	0.3834	1.3803
Mean Absolute Error	0.4727	1.7019

$$\text{Goodness of Fit} = 93.27\% \tag{3.12}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Vx_{est} - \text{mean}(Vx_{est}))}\right) \tag{3.13}$$

The model is tested even for real application on track.

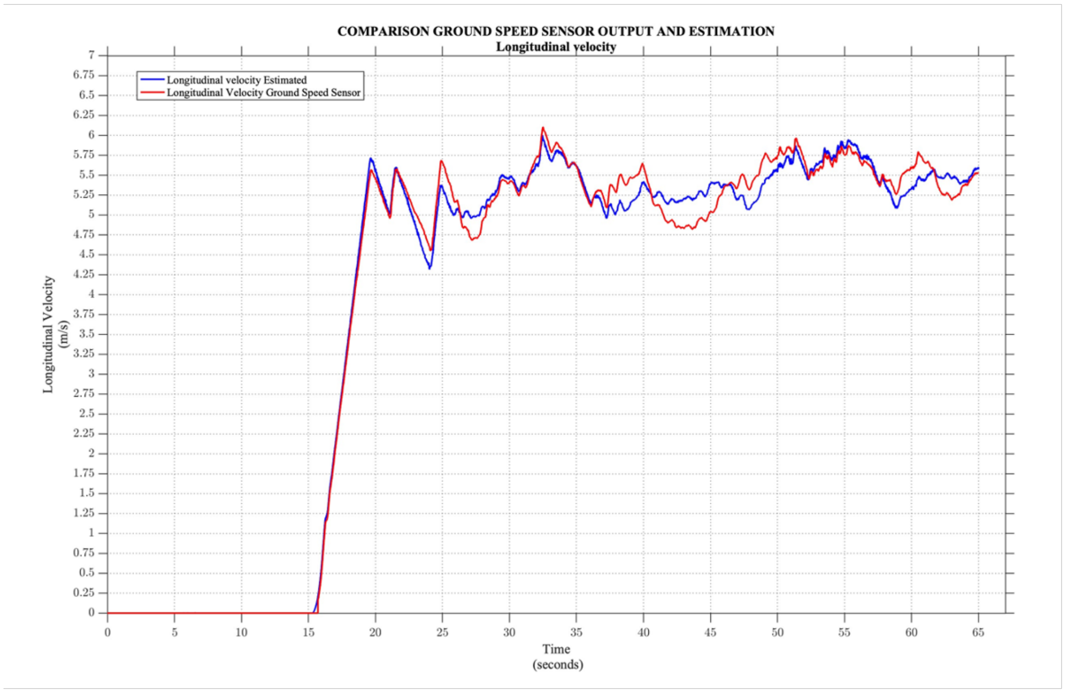

Figure 3.5: Longitudinal Velocity: Estimation vs Real Track Data

Table 3.4: Track Results: Longitudinal Velocity error

Error Type	Value in m/s	Value in km/h
Root Mean Square (RMS) Error	0.1585	0.5708
Mean Error	-0.0063	-0.0225
Mean Absolute Error	0.1141	0.4109

$$\text{Goodness of Fit} = 93.19\% \tag{3.14}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Vx_{est} - \text{mean}(Vx_{est}))}\right) \tag{3.15}$$

Chapter 4

Sideslip Angle Estimation

4.1 Adopted Strategy

As stated in Chapter 2, two different approach exist for the estimation of the sideslip angle of the vehicle. The first approach consists of the usage of Neural Networks and Empirical Models. Most of the methods adopted in literature have been introduced and the method proposed by Bonfitto et al.[10] has been explained.

The main issue related to the usage of Neural Networks is the fact that a lot of experimental Data are needed and that the vehicle should be available in advance. For this reason, this solution is not applicable for team purposes.

On the other hand the preferred solution is the usage of physical and mathematical models. In fact the main problems that usually are encountered for this solution are less influential in this kind of application respect to passenger cars. Cars parameters are well known and are almost constant (this is not true in passenger cars where mass and inertia change continuously due to different number of passengers and their weight). In autonomous race car this problem is not achieved.
Moreover, due to the impossibility of the usage of neural Network the usage of mathematical and physical models is mandatory.

As stated in Chapter 2, quoting [12], different versions of Kalman Filter are present for the estimation of the sideslip angle. The first one consists only of the usage of a kinematic Filter, the second only the usage of a Dynamic Filter and the third one a combination of both of them.

In this work, all the three strategies have been adopted,exploiting advantages and disadvantages of all of them, trying to find the best solution in simulation environment first and on real car subsequently.

A first attempt has been done with the Kinematic Filter, explained in this chapter. Chapter 5 is dedicated to the design of the dynamic filter in two different configurations (single-track model and pseudo dual-track model), while Chapter 6 describes the Combined Method and implementation on real car.

Before introducing the Kinematic Model a quick introduction on Kalman Filter is following.

4.2 Kalman Filter Introduction

At the current state of the art, different main approaches are used to estimate the vehicle sideslip angle by means of physical and mathematical models. Usually, in this kind of models, the time derivative of the requested quantity is calculated. However, the simple integral of this quantity is not enough to obtain good results, due to the presence of offsets, incorrect modeling, or uncertainties in some parameters that can lead to poor results and even instability. For this reason, a more sophisticated approach is needed. One of the most commonly used methods is the Kalman Filter [19] [20].

A Kalman filter is a mathematical algorithm and estimation tool used to estimate the state of a dynamic system from a series of uncertain measurements. It is widely employed in a diverse range of applications, including automotive, navigation, robotics, weather forecasting, telecommunications, and more. The Kalman filter is particularly useful when inferring the state of a system in real-time or in the presence of noise in measurements. Its primary goal is to provide an optimal estimate of the current state of the system, based on all past and present measurements, taking their uncertainties into account.

Here's how it works in a general sense:

1. **State Prediction**: Initially, the Kalman filter makes a prediction of the future state of the system based on the current state and the system's motion laws. This prediction also considers uncertainties associated with the system's motion.

2. **Measurement**: Subsequently, the filter receives a measurement of the system from the external world. However, these measurements are often subject to errors or noise.

3. **Update**: The filter combines the state prediction with the current measurement, assigning weights to them based on their relative uncertainties. This step is known as "update" and generates an improved estimate of the system's state.

The Kalman filter continually repeats these prediction and update steps as new measurements become available, allowing it to provide an increasingly accurate and reliable estimate of the system's true state over time.

Two main categories of Kalman filter are adopted. The first one is the Kinematic Kalman Filter. It is a mathematical model that does not consider the dynamic of the model, but just the kinematic relationship between different quantities. This kind of model is pretty useful when dynamic models are unknown, there is high uncertainty or variability in some parameters, and the quality of the sensors is very high and not susceptible to noise, offsets, and interferences.

On the other hand, in the Dynamic Kalman Filter, dynamic models are considered. The relationship between the quantities involved and their derivatives is not kinematic, but it is based on physical laws. This approach is widely used when the knowledge of all dynamic parameters (such as mass, inertia) is well known and does not vary over time. The time derivative of the quantities is no longer computed, but the evolution of the system is considered and estimated from the actual states and external inputs.

4.3 Kinematic Model

As stated previously, in the kinematic approach there is no dynamic modeling, and so the time derivative of the sideslip angle can be computed with the following formula:

$$\dot{\beta} = \frac{a_y}{V_x} - \frac{a_x}{V_x} \cdot \beta - \dot{\Psi} \cdot \beta^2 - \dot{\Psi} \tag{4.1}$$

In the previous formula, β is the vehicle sideslip angle estimated at the previous step, $\dot{\Psi}$ is the Yaw rate measured by the IMU sensor, a_y and a_x are respectively the lateral and the longitudinal components of the acceleration, and V_x is the longitudinal velocity of the vehicle estimated by the Fuzzy Logic as stated in Chapter 3.

With the assumption of small sideslip angles, the formula can be simplified as:

$$\dot{\beta} = \frac{a_y}{V_x} - \dot{\Psi} \tag{4.2}$$

with any consequences and significant variations of results.
Anyway, the general formula is adopted for the calculations of the derivative itself, while in the Kalman filter approach (next subsection), the state variable is not the sideslip angle β, but the lateral velocity of the vehicle, V_y, which, with reasonable assumptions, can be calculated as $V_x \cdot \beta$. This difference is due to the fact that the kinematic approach will be used in combination with the dynamic ones, implementing the fusion of the derivative of the sideslip angle (kinematic) with the sideslip angle itself (dynamic).

To avoid computational problems the derivative and the sideslip angle itself are forced to 0, if the estimated speed is lower than 1 m/s. It is reasonable, in fact to say that the slip is 0 if the speed is very low.

As stated in previous subsection, the derivative of the states is not enough to obtain good results, but it should be implemented in a mathematical model that considers both state prediction, but also measurements. Since a direct measurement of the sideslip angle is not present (there is not any sensor in the car that measures it) other state variables (dependent on the sideslip angle) should be predicted and compared with their measurement.

For this reason, a system of differential equations with two states will be employed in order to have a Kinematic Kalman Filter.

4.4 States and Prediction Phase

The evolution of two state variables is modeled (longitudinal and lateral components of the velocity). In fact, theoretically speaking, the time evolution of these two quantities can be modeled just using a kinematic model and the output of the sensors installed on the car.

The time evolution of the system can be described by the following set of equations:

$$\begin{cases} \dot{v} = -\dot{\Psi} \cdot u + a_y \\ \dot{u} = \dot{\Psi} \cdot v + a_x \end{cases} \tag{4.3}$$

In the previous equations v is the lateral component of the velocity, while u is the longitudinal one. The other values are directly taken as output of the Inertial Measurement Unit Sensor. The output of the system (to be compared with the measurement one) is the longitudinal component of the velocity.

The previous model can be seen as a Linear Time-Invariant System and can be described in matrix form as follows:

$$\dot{X} = A \cdot X + B \cdot U \tag{4.4}$$

$$Y = C \cdot X + D \cdot U \tag{4.5}$$

In the analyzed system, the matrices are identified as follows:

State matrix A:

$$A = \begin{bmatrix} 0 & -\dot{\Psi} \\ \dot{\Psi} & 0 \end{bmatrix} \tag{4.6}$$

Input Matrix B:

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.7}$$

Input Vector U:

$$U = \begin{bmatrix} a_x \\ a_y \end{bmatrix} \tag{4.8}$$

Output Vector C:

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.9}$$

D = 0

This is true if we consider the system as continuous in the time domain. However, since all the sensor outputs are discrete due to quantization in time, and the computation time is not negligible, for real-time implementation where each iteration lasts 0.005 s, the model should be discretized, and some modifications are necessary. The system can be modeled after a few mathematical processes that are not described as follows:

$$X_t = (A \cdot \Delta t + I) \cdot X_{t-1} + (\Delta t \cdot B) \cdot U_{t-1} \tag{4.10}$$

$$Y = C \cdot X + D \cdot U \tag{4.11}$$

In the previous set of equations, A, B, C, and D are the matrices previously described, while I is the Identity Matrix, and Δt is the sampling time. The new values of the state variables are calculated and updated.

4.5 Measurement and Update

The second step of the Kalman Filter algorithm is the measurement phase. The output is calculated as:

$$Y = C \cdot X + D \cdot U \tag{4.12}$$

In this model, it corresponds to the longitudinal component of the velocity, and it is compared to the measured one. In this case, the correct value of the velocity (since a ground speed sensor is not present) is assumed equal to the one estimated by the Fuzzy Logic.

At this point, the State Covariance Matrix "P" is updated. It is a fundamental component of the Kalman filter and is used to quantify the uncertainty associated with the estimated state of the system. The P matrix keeps track of the covariances between various state variables of the system and is updated iteratively during the Kalman filter's prediction and update process to reflect the evolution of uncertainties over time. At the first iteration, the matrix P is initialized with the following values:

$$P = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \tag{4.13}$$

Also, the Process Covariance Matrix and Measurement Covariance Matrix are initialized as follows (they will not be changed during the iterations):
Process Covariance Matrix Q:

$$Q = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \tag{4.14}$$

Measurement Covariance Matrix R:

$$R = 10^{-10} \tag{4.15}$$

The choice of these values is compatible with most common values in this kind of application. During the update phase, the Matrix "P" is updated according to the following method:

$$P_t = (A\Delta t + I)P_{t-1}(A\Delta t + I)^T + Q \tag{4.16}$$

The Kalman Gain Matrix can be computed. This matrix is fundamental to the Kalman filter's update process. The Kalman Gain Matrix determines how much weight to give to the state prediction versus the current measurement during the state estimation update. Its proper selection is crucial for obtaining an optimal estimate of the system state. It is calculated dynamically at each Kalman filter update step and depends on the covariances of the prediction error and measurement error. The Kalman Gain "K" is computed as follows:

$$K = P \cdot C^T \cdot (C \cdot P \cdot C^T + R)^{-1} \tag{4.17}$$

At this point, the state variables can be updated, applying the correction due to the difference with respect to the estimated and measured state:

$$X = X + K \cdot (Y - Y_{\text{est}}) \tag{4.18}$$

where Y is the measured longitudinal component of the velocity, and Y_{est} is the output of the LTI system. The last step consists of the new update of the matrix P as follows:

$$P = P - K \cdot C \cdot P \tag{4.19}$$

At this point, all the steps of the Kalman Filter algorithm are completed. This method is fundamental to avoid divergences due to offset and disturbances of sensor.

Anyway, since the only measured state is the longitudinal component of the velocity is slightly dependent on the lateral velocity (the absolute value of the velocity is quite low), a Kinematic Kalman Filter does not represent the best solution for this kind of applications.

Moreover, since the value of the velocity is not directly measured, but it is still estimated by a fuzzy logic this kind of solution do not show good results, even in simulation environment. In most of common application, especially in hard cornering conditions, anyway in common uses the Kinematic Kalman Filter can show good results and can be combined with the Dynamic one that will be explained in following sections.

4.6 Results

In this section the results obtained in simulation with the Kinematic Filter are discussed.

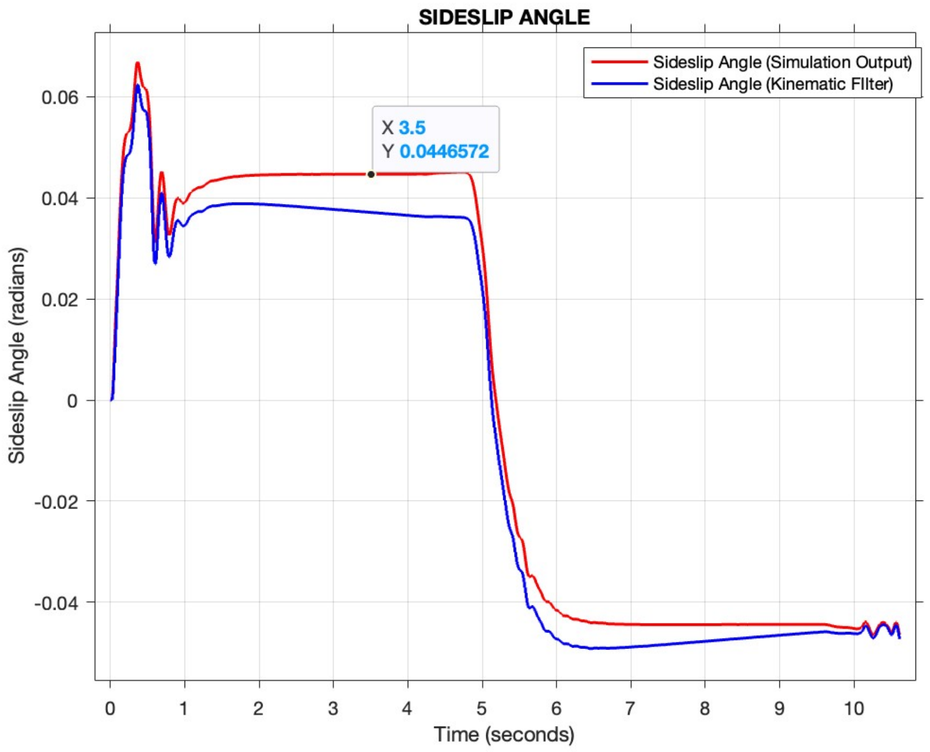

Figure 4.1: Sideslip angle: Kinematic Kalman Filter vs Estimation

In fig4.2 the sideslip angle obtained by means of the Kinematic Filter is compared with the sideslip angle taken as Vi-Grade simulation output.

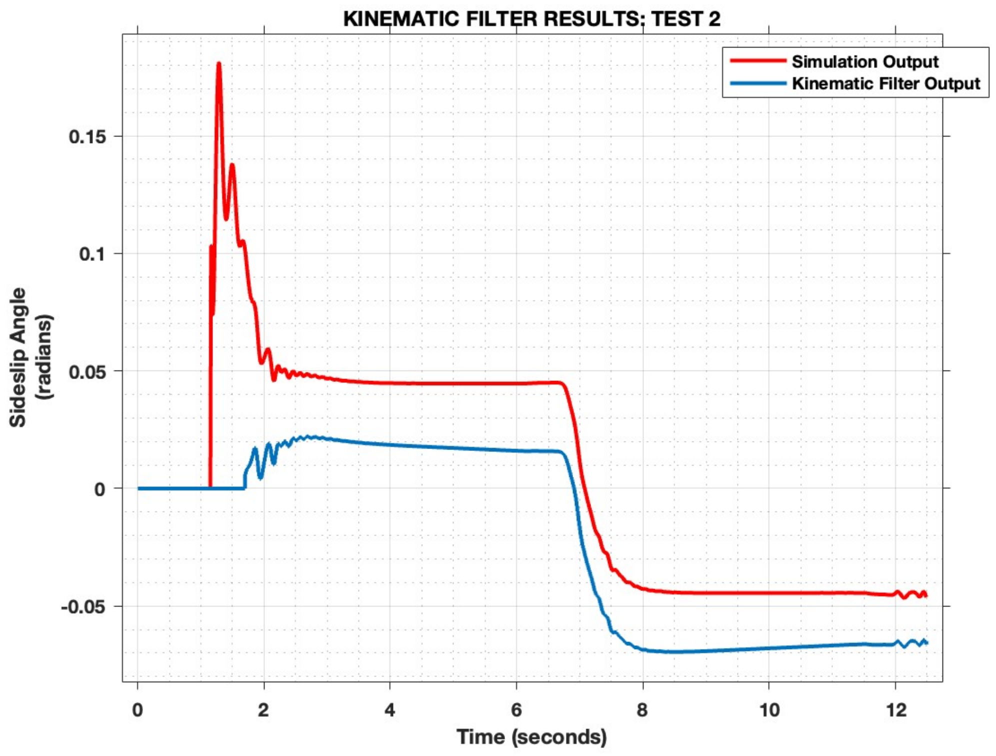

Figure 4.2: Sideslip angle: Kinematic Kalman Filter vs Estimation. Test 2

Another simulation has been performed (same approach of chapter 3) with initial velocity equal to zero. In this case the convergence of the model to the correct values is not guaranteed as shown in the next figure:

In the following table the results are reported.

Table 4.1: Simulation Results: Sideslip Angle error: Kinematic Filter

Error Type	Value in radians	Value in degrees
Root Mean Square (RMS) Error	0.0450 rad	2.5785 deg
Mean Error	-0.0215 rad	-1.2333 deg
Mean Absolute Error	0.0359 rad	2.0568 deg

$$\text{Goodness of Fit} = 43.06\% \tag{4.20}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Beta_{est} - \text{mean}(Beta_{est}))}\right) \tag{4.21}$$

As it is possible to notice the results are not so pretty good. In fact, even if the behavior of the sideslip angle is perfectly followed by the kinematic estimator, the error is not equal to 0.

There are multiple reasons that lead to this.The main reason is the fact that the estimated output of the Kalman (longitudinal velocity) is the only factor that leads to correction.

Firstly, as it is possible to notice from the equations (4.1) of the Kalman Filter the dependance of the derivative of the longitudinal velocity respect to the lateral one is much lower than the contribute of the longitudinal acceleration. Moreover, the real value of the velocity is not known, but it is taken as the output of the fuzzy logic estimator described in chapter 3.

Anyway, as previously introduced, even if not satisfactory for team purposes the behavior of the sideslip angle evolution is well followed by the estimator. The fig4.3 represents the time derivative of the sideslip angle (estimation and simulation output).

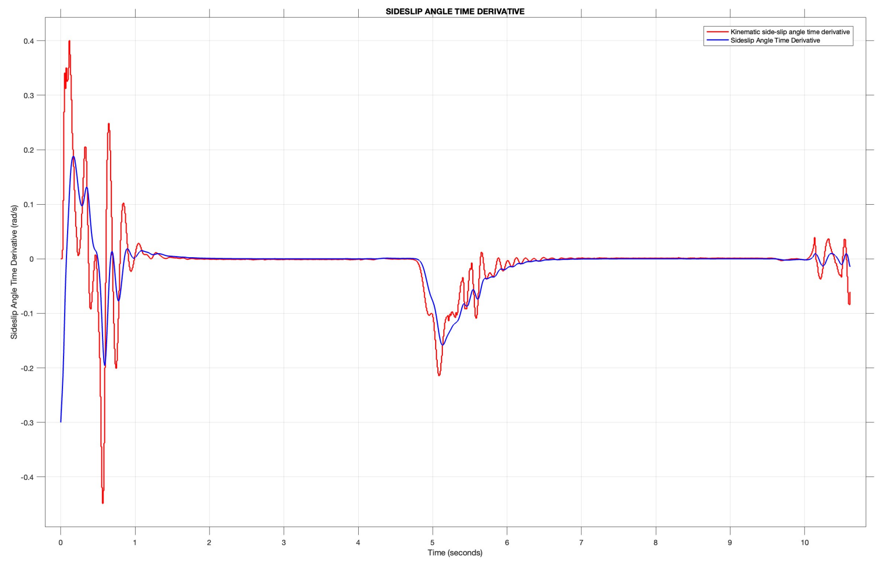

Figure 4.3: Sideslip angle Derivative: Kinematic Kalman Filter vs Estimation

As it is possible to notice, except for initial mistakes (due to low uncorrect speed), the derivative of the sideslip angle is estimated very well. For this reason in next chapters a way to use it will be discussed.

The previous graphs have been obtained with the experiment with initial speed equal to zero, but they are comparable with the ones obtained in the other simulation, with initial speed different than 0.

Chapter 5

Dynamic Approach

5.1 Vehicle Dynamics Modelling

At the current state of the art, various vehicle dynamics models with different assumptions and degrees of freedom exist, depending on the model's complexity, the quality of results required, computational power needed, and the type of application. Since this work is related to the sideslip angle, different models of lateral dynamics will be analyzed.

Many models consider the vehicle's movement on its own suspensions, resulting in a minimum of 10 degrees of freedom [21]. Fortunately, for many applications, it is not necessary to use such a complex model, and a rigid model [22] is sufficient. The rigid model does not consider the movement of the vehicle body on the suspensions. Therefore, we look at the problem from a top view, neglecting the vertical dynamics of the vehicle.

To use the rigid vehicle model, it is necessary to define two reference systems. The first is the inertial reference frame OXY, which can be defined with arbitrary orientation, and the second is the local reference frame Gxy, which is oriented consistently with the vehicle. The angle ψ, called the yaw angle, is the angle between the X axis of the global reference frame and the longitudinal plane of the vehicle. The angle β, called the vehicle sideslip angle, is the angle between the velocity vector of the vehicle and the local longitudinal (x) axis. The sideslip angle can be computed starting from the components of the velocity vector in the two directions of the local reference frame:

$$\beta = \frac{v_y}{v_x} \tag{5.1}$$

5.2 Tire Behavior and Forces

As stated in previous sections, longitudinal and lateral forces are exchanged by the vehicle and the road. The way in which forces are exchanged depends mainly on tires and road conditions.

During the years different ways to model the tire behavior accordingly to different road and driving conditions have been developed. The modelling of a tire [23] can generally be performed using two distinct procedures: one involves creating a physical-theoretical model aimed at physically justifying and quantifying the phenomena that affect tire dynamics; the other involves creating an empirical-mathematical model with the purpose of replicating the characteristic behaviors of the real component based on mathematical formulas created specifically following experimental characterizations, independent of the physical reality that determines the behavior acquired through measurements.

An empirical-mathematical model is typically less complex and easier to integrate into vehicle dynamics models, while a physical-theoretical model, starting from a physical study and therefore based on laws and equations that seek to represent reality, is more complex but also potentially suitable for conducting a detailed analysis of tire performance in relation to design parameters. An example of a physical-theoretical model is the brush model [24], which can physically justify the phenomena that generate tire-to-ground force exchanges. The empirical-mathematical model to be presented is one of the most important, perhaps the most widespread in the field of vehicle dynamics and is implemented in two of the three tire models used for simulations in this thesis. It is known as the "Pacejka Magic Formula [25]," introduced in 1987 by H.B. Pacejka.

Therefore, the "Pacejka Magic Formula" is an empirical-mathematical model that seeks to summarize the experimental performance of the tire through mathematical formulas. These formulas have a specific structure in which coefficients are quantified based on specific experimental tests.

In general, these curves allow obtaining the trends of the actions determined by the brush model:

- Longitudinal force F_x
- Lateral force F_y
- Self-aligning torque M_z

as functions of longitudinal slip s, slip angle α, and camber angle γ.

Some concepts regarding longitudinal force exchange have been already introduced in vehicle velocity estimation chapter. Modern Pacejka Models introduced the combined behavior [26] between longitudinal and lateral dynamics (i.e. for example the same tire produces different lateral forces in the same load and sideslip conditions if a longitudinal slip is present or not). In this work some assumptions have been done. The longitudinal and lateral interaction has been neglected and all the lateral forces will be calculated as if the longitudinal slip was null.

Another important assumption is the fact that the camber angle does not contribute to lateral forces. This is obviously not true because a not negligible contribute of the development of the lateral force of a single tire comes from the camber angle. Anyway, in first assumption a single-track model has been developed in order to estimate the lateral force of each axle. Since each axle is composed by two wheels and since in steady conditions the left wheel camber angle is equal to the opposite one of the right one, an initial value of the camber angle must be set to 0 in this kind of model. During different driving conditions, the architecture of the suspension and load transfer can modify the camber angle of the wheels, leading to a non-zero resultant value. As introduced in previous subsections, vertical motion of the vehicle is not modelled and the value of the camber angle of each wheel in all driving conditions is practically unknown. For this reason, the camber angle effects, at least in this initial phase, are neglected.

At this point it is possible to predict the lateral forces exchanged by the tires accordingly to the load and the sideslip angle. In single-track modelling it is assumed that each axle is composed by one single-wheel with a load equal to the sum of the two wheels of the axle. Lots of coefficients are involved in the empirical model formulation of the forces exchanged by the tires. Since only the lateral forces are of interest, the coefficients related to lateral forces estimation (given by Pirelli) are reported.

PARAMETER	**VALUE**	**INFLUENCE**
pcy1	1.0994	Influence the stiffness with nominal load
pDy1	1.7067	Influence the stiffness variation due to load variation
pDy2	-0.4109	Introduce non-linearities due to load variation
pDy3	0	Effects of camber angle (not used)
pky1	-40.95	Influence of non-linearities of the model
pky2	1.1463	Influence of non-linearities of the model (load effects)
pky3	0.20173	Non-linearities due to camber (not used)
pEy1	0.22355	Influence of sideslip angle
pEy2	0.6748	Non-linearities of sideslip angle (load variation)
pEy3	1.182	Non-linearities of sideslip angle (sideslip variation)
pExy4	0.037357	Influence of longitudinal slip (not used)
Fz0	1000	Nominal load of the wheel

Table 5.1: Pacejka Parameters

At this point it is possible to estimate the side forces exchanged by the wheels accordingly to their load and sideslip angle. How load and sideslip are estimated is reported in next section (Extended Kalman Filter). Firstly, the difference between the load acting on the wheel and the nominal one (1000 N) is computed.

$$dfz = \frac{Fz - 1 \cdot Fz0}{1 \cdot Fz0} \tag{5.2}$$

Then the friction coefficient between tire and road is estimated.

$$\mu_{py} = pDy1 + pDy2 \cdot dfz \tag{5.3}$$

Then five different variables B, C, D, and E are computed in the following way:

$$Kya0 = Fz0 \cdot pky1 \cdot \sin(2 \cdot \arctan(Fz/(pky2 \cdot Fz0)) \tag{5.4}$$

$$Kya = Kya0 \cdot (1 - pky3 \cdot (\sin(\gamma))^2) \tag{5.5}$$

$$E_y = (pEy1 + pEy2 \cdot dfz) \cdot \big(1 - \big(pEy3 + pExy4 \cdot (\sin(\gamma))^2 \cdot \text{sign}(\alpha)\big)\big) \tag{5.6}$$

$$Dy = \mu_{py} \cdot Fz \tag{5.7}$$

$$B_y = \frac{K_{ya}}{C_y \cdot D_y} \tag{5.8}$$

The force acting on the single wheel is finally estimated in the following way.

If a single track model approach is considered the force acting on the axle is calculated with the following formula:

$$F_{yst}(i) = D_{y_r} \cdot \sin(C_y \cdot \arctan\left(B_y \cdot (1 - E_y) \cdot \alpha_r + E_y \cdot \arctan(B_y \cdot \alpha)\right)) \tag{5.9}$$

while in the case of double track the force is estimated in the following way (the front axle formulas are shown but they are valid also for rear axle):

$$F_{yfl} = D_{yfl} \cdot \sin(C_y \cdot \arctan\left(B_{y_{fl}} \cdot (1 - E_{yfl}) \cdot \alpha_f + E_{yfl} \cdot \arctan(B_{yfl} \cdot \alpha_f)\right)) \tag{5.10}$$

$$F_{yfr} = D_{yfr} \cdot \sin(C_y \cdot \arctan\left(B_{y_{fr}} \cdot (1 - E_{yfr}) \cdot \alpha_f + E_{yfr} \cdot \arctan(B_{yfr} \cdot \alpha_f)\right)) \tag{5.11}$$

$$F_{yf} = F_{yfr} + F_{yfl} \tag{5.12}$$

A graph reporting the forces exchanged by the wheel with different load conditions and sideslip angle is reported in order to evidence all the non-linearities of the model.

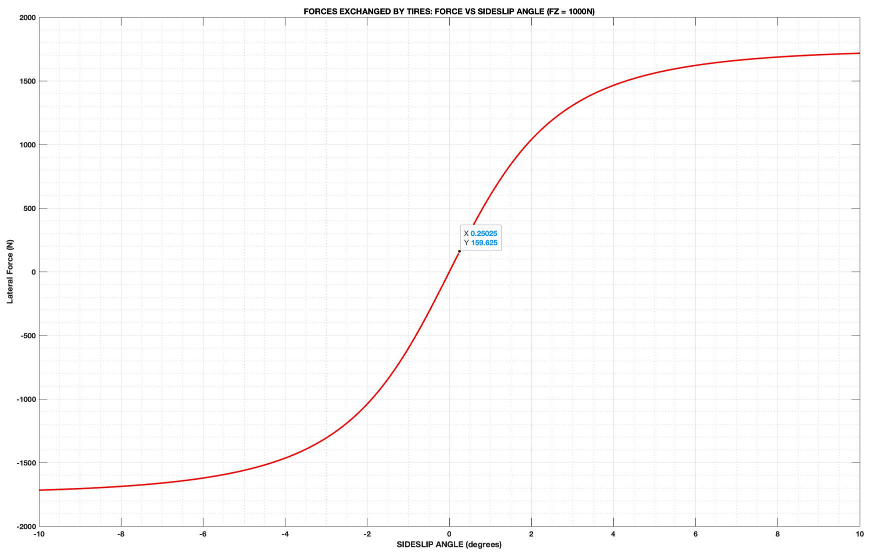

Figure 5.1: Lateral Force vs Sideslip Angle

As it is possible to notice, for small values of sideslip angle the relationship between force and sideslip angle is almost linear and the model can be approximated by a straight line. The pendency of this straight is called Cornering Stiffness. It quite intuitive to say that bigger loads lead to higher cornering stiffness. The following plot evidences the relationship between cornering stiffness and load.

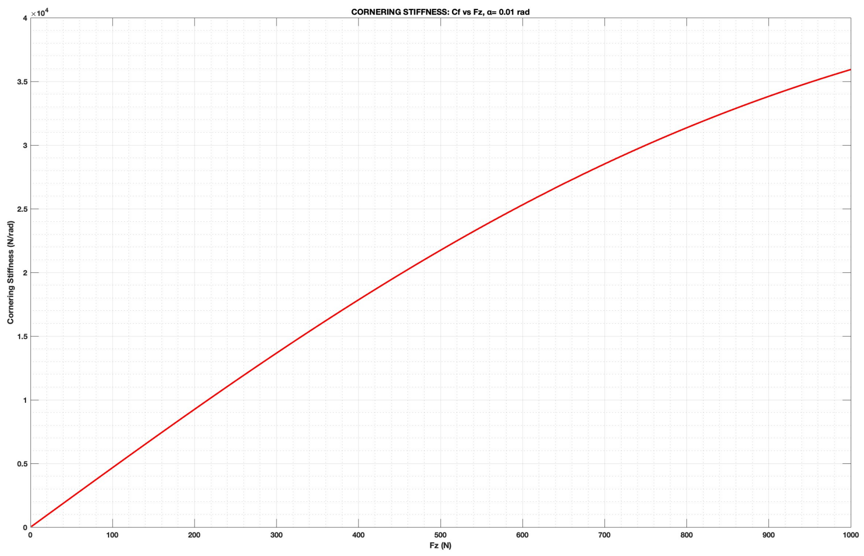

Figure 5.2: Cornering Stiffness vs Load

As it is possible to notice the relationship is not linear (it is quite linear only for small values of Fz). Since the load acting on each axle is usually in non-linear region, a single-track model can be not adapted for this kind of application, as it is not good to estimate the correct cornering stiffness of the axle. For this reason, another model that takes more in account all the nonlinearities is proposed.

A complete model present in literature is the so called dual-track model. This kind of model assume that the vehicle movement is planar motion, and other vehicle motions such as pitch, roll, and vertical motions are also neglected. When the longitudinal motion is added and the dynamics for four wheels are also addressed for the single-track model, causing the four-wheel vehicle model to then involve longitudinal and lateral motions and yaw motion, the so-called double-track model is often used to estimate longitudinal and lateral states. The equations are reported:

$$\begin{cases} F_x = m \cdot (v_x \dot{v_y} - \dot{v_y} \cdot \dot{\psi}) \\ F_x = m \cdot (v_x \cdot \dot{\psi} + \dot{v_y}) \\ J_z \ddot{\psi} = M_z \end{cases} \tag{5.13}$$

$$F_x = \sum_i F_{x,i} \cos\delta_i - \sum_i F_{y,i} \sin\delta_i - \frac{1}{2}\rho\, S\, C_x\, V^2 - m\, g \sin\alpha_{rg}$$

$$F_y = \sum_i F_{x,i} \sin\delta_i + \sum_i F_{y,i} \cos\delta_i + \frac{1}{2}\rho\, S\, C_y\, V^2 + F_{y,e}$$

$$M_z = \sum_i (F_{x,i} \sin\delta_i)\, x_i + \sum_i (F_{y,i} \cos\delta_i)\, x_i - \sum_i (F_{x,i} \cos\delta_i)\, y_i + \sum_i (F_{y,i} \sin\delta_i)\, y_i + \sum_i M_{z,i} + \frac{1}{2}\rho\, S\, l\, C_{M_z}\, V^2 + M_{z,e}$$

Figure 5.3: Dual Track Forces and Moments

The explained model considers all the forces exchanged. Moreover, in the lateral dynamics are present longitudinal forces and vice versa, in longitudinal dynamics takes into account lateral forces. In this work a simpler model is proposed. The lateral forces exchanged by each wheel are considered, but the relationship between longitudinal forces and lateral dynamics is neglected. Moreover, aerodynamics forces are neglected. More details are present in the following section. In this way, since the cornering stiffness of an axle is not computed as if all the load was distributed on a single wheel, but it is given by the sum of the cornering stiffness of the two wheels that belongs to that axle, the nonlinear relationship between cornering stiffness and load is considered. The differences are evidenced in the following graph.

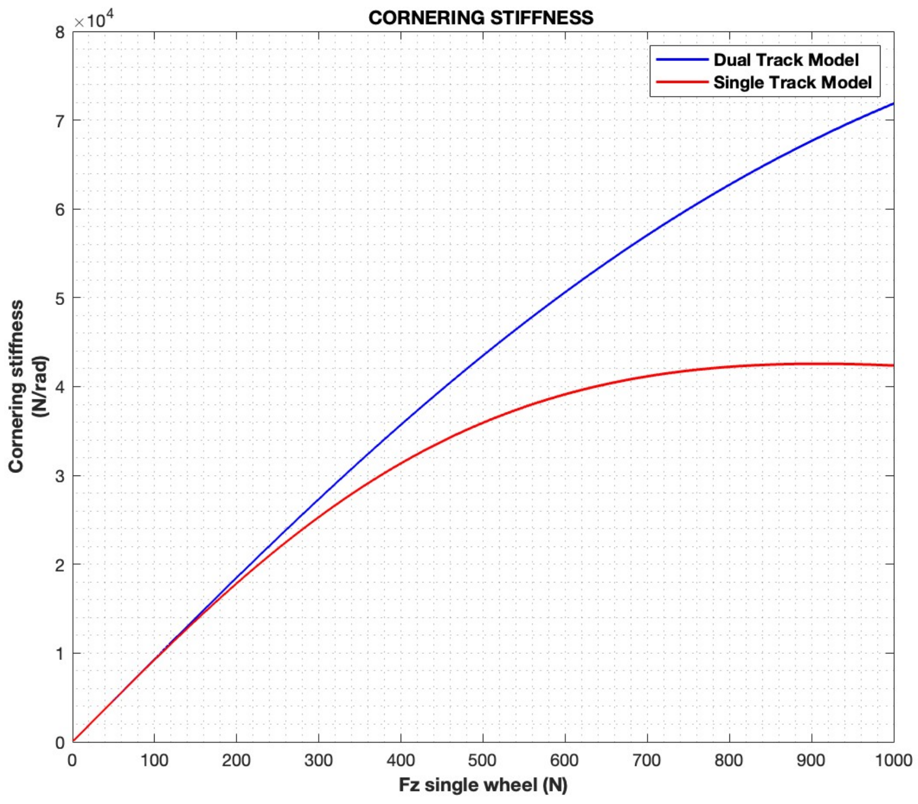

Figure 5.4: Cornering Stiffness: Single-Track Model vs Dual Track Model

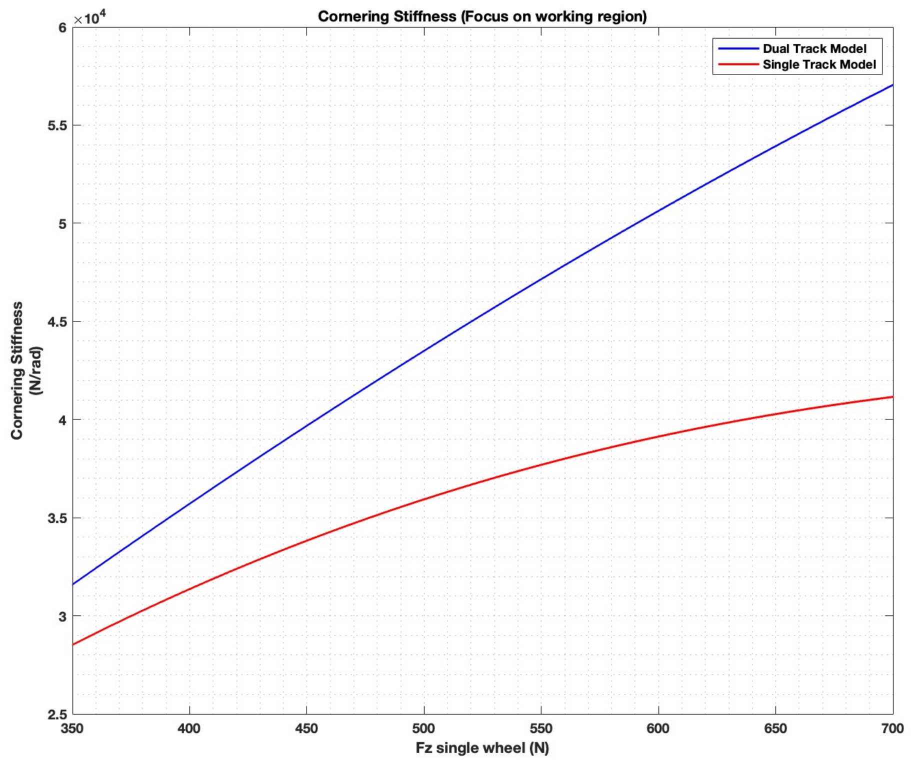

Figure 5.5: Cornering Stiffness: Single-Track Model vs Dual Track Model. Focus On Working Region

5.3 Extended Kalman Filter

5.3.1 Dynamics Model

The Kalman Filter approach and method was well explained in the chapter dedicated to the Kinematic Kalman Filter. In this section its adaptation to a Dynamic Model is proposed.

The main difference between Kinematic and Dynamic approach consists of the fact that in Kinematic approach the relationship between different variables is purely kinematic and does not take into account any physical law, while in Dynamic approach the time evolution of each variable and state is calculated accordingly to mathematical and physical Laws (such as Newton Laws, Thermodynamic Laws, Euler Equations etc.)

In Dynamic approach, the usage of mathematical and physical Laws requires the knowledge of all the parameters involved and assumptions in which laws are valid or can be simplified are usually required. In some cases when this is not possible, Dynamics Models are substituted by Metamodels (as in the case of tires behavior).

In this work, a Dynamic Model for Vehicle Motion is introduced. As stated in previous section different models with different complexity and accuracy are present in literature and at the actual State of the Art.

Complex Simulators, such as Vi-Grade (a software used in this work) take into account a very big number of degrees of freedom (they do not just consider vehicle states of motion but also the time evolution of each component state such as wheel speed, differential speed, motors speed is considered). For this kind of application, the knowledge of a very big number of parameters (such as components stiffness, weight, and Inertia) is needed. In simulators like this, also effects of tire consumption and temperature and transfer functions are considered and the quality of the results of this kind of application can be very high if the correctness of parameters and models is ensured.

Simpler models, but with remarkable accuracy, neglect effects of driveline dynamics, transfer functions, tire consumption and temperature effects. These models consider usally called 10 degrees of freedom models consider the relationship between different state of variables (including vertical motion, pitch, heave etc.)

The equations are reported into the following image:

$$
\begin{cases}
① & m\dot{V} + J_S\ddot{\theta} = \sum_i F_{x,i} - \frac{1}{2}\rho S C_x V^2 \\
② & mV(\dot{\beta} + \dot{\psi}) - m_S h\ddot{\varphi} - \sum_i m_i q_i \ddot{\varphi}_i = Y_\beta \beta + Y_{\dot{\psi}}\dot{\psi} + Y_{\dot{\varphi}}\dot{\varphi} + \sum_i Y_{\dot{\varphi}_i}\dot{\varphi}_i + Y_\delta \delta + F_{y,e} \\
③ & m_S\ddot{Z} - m_S c\ddot{\theta} + \sum_i c_i\dot{Z} - \sum_i c_i x_{a,i}\dot{\theta} + \sum_i k_i Z - \sum_i k_i Z_i - \sum_i k_i x_{m,i}\theta = \frac{1}{2}\rho S (C_Z)_\theta(\theta + \theta_0)V^2 + \sum_i k_i L_i - m_S g \\
④ & m_i\ddot{Z}_i + (c_i + 2c_{p_i})\dot{Z}_i - c_i\dot{Z} + c_i x_{a,i}\dot{\theta} + (k_i + 2P_i)Z_i - k_i Z + k_i x_{m,i}\theta = -k_i x_{m,i}\theta_o - k_i L_i + 2P_i L_{p,i} - m_i g \\
⑤ & J_Z\ddot{\psi} - J_{xZ}\ddot{\varphi} + \dot{V}\left(m_S h\varphi + \sum_i m_i q_{0,i}\varphi_i\right) - \sum_i J_{xZ_i}\ddot{\varphi}_i = N_\beta \beta + N_{\dot{\psi}}\dot{\psi} + N_\varphi \varphi + \sum_i N_{\varphi_i}\varphi_i + N_\delta \delta + M_{Z,e} \\
⑥ & J_y\ddot{\theta} + J_S\dot{V} - m_S c\ddot{Z} - \sum_i c_i x_{a,i}\dot{Z} + \sum_i c_i x_{a,i}\dot{\theta} + \sum_i c_i x_{a,i}\dot{Z}_i - \sum_i k_i x_{m,i} Z + \sum_i k_i x_{m,i}^2\theta + \sum_i k_i x_{m,i} Z_i - m_S g h\theta = \\
 & = -\sum_i k_i x_{m,i} L_i + M_\theta \theta + \frac{1}{2}\rho S\left[hC_x - \left(C_{M_y}\right)_\theta \theta\right]V^2 + \sum_i F_{x,i}(q_{i,0} + Z_{iC}) + m_S g(h\theta_0 + c) \\
⑦ & J_x\ddot{\varphi} - J_{xZ}\ddot{\psi} - m_S h(\dot{v}_y + V\dot{\psi}) = L_\beta \beta + L_\varphi \varphi + L_{\dot{\varphi}}\dot{\varphi} + \sum_i L_{\varphi_i}\varphi_i + \sum_i L_{\dot{\varphi}_i}\dot{\varphi}_i \\
⑧ & J_{x_i}^*\ddot{\varphi} - J_{xZ_i}^*\ddot{\psi} - m_i q_{i,0}(\dot{v}_y + V\dot{\psi}) = L_{\beta,i}\beta + L_{\dot{\psi},i}\dot{\psi} + L_{\varphi,i}\varphi + L_{\dot{\varphi},i}\dot{\varphi} + L_{\varphi_i,i}\varphi_i + L_{\dot{\varphi}_i,i}\dot{\varphi}_i + L_{i,\delta}\delta
\end{cases}
$$

Figure 5.6: 10 Degrees of Freedom Equations

Please notice, that in the above set of equations only 8 equations are reported and not 10. Anyway, due to the fact that symbols with letter "i" refer to both front and rear axle and so the number of variables is equal to 10.

As it is possible to notice, for this kind of model the knowledge of a lot of parameters (aerodynamics, suspensions stiffness and damping, position of center of mass, derivatives of stability) are absolutely needed. Anyway, models like this cannot be implemented in Driverless applications due to high computational power consumption and their strong dependance to unknown parameters.

With some assumption (if the longitudinal velocity and acceleration are known) it is possible to further simplify this set of equation, considering only equations related to handling:

$$\begin{matrix} \textit{Handling} \\ \beta,\psi,\varphi,\varphi_i \end{matrix}\begin{cases} mV(\dot\beta+\dot\psi) - m_S h\ddot\varphi - \sum_i m_i q_i \ddot\varphi_i = Y_\beta\beta + Y_{\dot\psi}\dot\psi + Y_{\dot\varphi}\dot\varphi + \sum_i Y_{\dot\varphi_i}\dot\varphi_i + Y_\delta\delta + F_{y,e} \\ J_Z\ddot\psi - J_{xZ}\ddot\varphi + \dot V\left(m_S h\varphi + \sum_i m_i q_{0,i}\varphi_i\right) - \sum_i J_{xZ_i}\ddot\varphi_i = N_\beta\beta + N_{\dot\psi}\dot\psi + N_\varphi\varphi + \sum_i N_{\varphi_i}\varphi_i + N_\delta\delta + M_{Z,e} \\ J_x\ddot\varphi - J_{xZ}\ddot\psi - m_S h(\dot v_y + V\dot\psi) = L_\beta\beta + L_\varphi\varphi + L_{\dot\varphi}\dot\varphi + \sum_i L_{\varphi_i}\varphi_i + \sum_i L_{\dot\varphi_i}\dot\varphi_i \\ J^*_{x_i}\ddot\varphi - J^*_{xZ_i}\ddot\psi - m_i q_{i,0}(\dot v_y + V\dot\psi) = L_{\beta,i}\beta + L_{\dot\psi,i}\dot\psi + L_{\varphi,i}\varphi + L_{\dot\varphi,i}\dot\varphi + L_{\varphi_i,i}\varphi_i + L_{\dot\varphi_i,i}\dot\varphi_i + L_{i,\delta}\delta \end{cases}$$

Figure 5.7: Handling equations

This simplification is possible since the speed is not estimated with dynamics model and the longitudinal acceleration is directly measured by Inertial Measurement Sensors.

A further simplification is possible since the natural frequency of sprung and unsprang of mass differs of one order of magnitude. For this reason, it is possible to neglect the roll motions of each axle, obtaining the following set of equations, called Segel Model:

$$\textbf{\textit{Segel model}}\quad \begin{cases} mV(\dot\beta+\dot\psi) - m_S h\ddot\varphi = (Y_\beta - m\dot V)\beta + Y_{\dot\psi}\dot\psi + Y_\varphi\varphi + Y_\delta\delta + F_{y,e} \\ J_Z\ddot\psi - J_{xZ}\ddot\varphi + m_S h\dot V\varphi = N_\beta\beta + N_{\dot\psi}\dot\psi + N_\varphi\varphi + N_\delta\delta + M_{Z,e} \\ J_x\ddot\varphi - J_{xZ}\ddot\psi - m_S hV(\dot\beta+\dot\psi) = (L_\beta + m_S h\dot V)\beta + L_\varphi\varphi + L_{\dot\varphi}\dot\varphi \end{cases}$$

Figure 5.8: Segel Model Equations

Anyway, this kind of model is still too complex for Formula Student Driverless Application due to its dependency to unknown ad not measurable quantities and variables. For this reason, a Rigid Model that considers two equations and two variables is proposed.

In rigid models, all vertical dynamics is neglected and the vehicle is assumed to be rigid: so all deformations and roll motions are neglected.

The states equation of the Kalman Filter are reported:

$$\begin{cases} \dot\beta = -\dot\psi + \frac{F_{y_f}+F_{y_r}}{m\cdot V_x} \\ \ddot\psi = \frac{F_{y_f}\cdot l_F}{J_z} - \frac{F_{y_r}\cdot l_R}{J_z} + \frac{M_{z\,\text{applied}}}{J_z} \end{cases} \tag{5.14}$$

This set of equations with two variables (sideslip angle and Yaw Rate) considers the variable that most influence the Sideslip angle and it is an optimum compromise between model complexity, results quality, and knowledge of parameters.

As it is possible to see the necessary parameters are vehicle mass, center of gravity position and vehicle inertia. All of them are not directly measured but are estimated accordingly to CAD design projects and so they represent a source of error, due to the high uncertainty of their real values.

5.3.2 Measurement and Update Phase

Once the state prediction phase is concluded (the new values of sideslip angle and yaw rate are calculated accordingly to the state equations) it is possible to estimate the lateral force exchanged by each axle (in first analysis a single-track model described in previous sections it is used).

As already stated, the lateral forces depend on load conditions and tire's sideslip angle. Anyway, its dependence on the sideslip angle and yaw rate is not null because the tire sideslip angle (α) is strictly dependent on other variables, and it is computed in the following way for front and rear axle.

$$\begin{aligned} \alpha_f &= -\left(\delta - \frac{l_F \cdot \dot{\psi}}{V_x}\right) + \beta \\ \alpha_r &= -\left(-\beta + \frac{l_R \cdot \dot{\psi}}{V_x}\right) \end{aligned} \tag{5.15}$$

It is important to notice that in this formula, the convention is opposite to the one of the conventional reference frame, but α is considered positive if in clockwise direction, while in the main reference frame, α is positive in counterclockwise direction.

In this way it is possible to estimate the lateral forces exchanged by each axle. The real value of the forces, obviously, is not known because a direct measurement of this quantity is impossible.

Anyway, these quantities can be measured indirectly. In fact, since a direct measurement of the lateral acceleration is possible through IMU sensor it is possible to calculate the sum of the lateral forces of the two axles (II Newton Law). This is possible, applying the assumptions made in previous sections (longitudinal force effect on lateral dynamics is neglected). Anyway, the sum of the forces exchanged by the two axles does not represent a good measure, since the value of the force of each axle is not calculated.

For this reason, another approach should be found in order to measure indirectly the lateral forces exchanged by each axle. Inertial Measurement Units sensors do not measure only accelerations, but also Yaw Rate (taking advantage of the Coriolis Force). This quantity does not strictly depend on lateral forces, but its derivative does it, so applying a discrete-time derivative to this measured quantity it is possible to calculate the correct values of the lateral forces in the following way:

$$\begin{cases} Fy_f = \dfrac{1}{\cos(\delta)}\left(\dfrac{m \cdot a_y \cdot l_R + J_z \cdot \dot{Y}_r - u_k}{l_F + l_R}\right) \\ Fy_r = \dfrac{m \cdot a_y \cdot l_F - J_z \cdot \dot{Y}_r + u_k}{l_F + l_R} \end{cases} \tag{5.16}$$

Now it is possible to compare the estimated values of Yaw Rate, Front axle Force and Rear axle Force respect to their measured values. At this point, it is possible to calculate the difference between measured and estimate quantities and so it is possible to calculate the error:

$$\begin{cases} \text{err}(1) = \dot{\psi}_{\text{meas}} - \dot{\psi}_{\text{est}} \\ \text{err}(2) = Fy_{f_{\text{meas}}} - Fy_{f_{\text{Pacejka}}} \\ \text{err}(3) = Fy_{r_{\text{meas}}} - Fy_{r_{\text{Pacejka}}} \end{cases} \tag{5.17}$$

The next steps are very similar to the ones described in kinematic approach, even if they are slightly different and so they are now briefly described.

Firstly, the Measurement Matrix H is calculated. This Matrix represents the derivatives of the measured state (Yaw Rate, Fy_f and Fy_r) with respect to the state variables (β and Ψ).

In this way a 3x2 matrix is obtained, where the (i, j) coefficient represents the value of the derivative of the (i) measured state respect to the (j) state variable. The Matrix is reported in the Appendix.

The equations are reported in the single track model. In fact, the coefficient of the axles are reported and not the ones of the single tire. The matrix are exactly the same for the dual track model, taking into account that the force of each tire is computed.

Now the Jacobian Matrix “F” is calculated. This Matrix represents the derivatives of the state variables time derivative respect to the state variables themselves. The values of each matrix coefficients are reported in the Appendix.

The Matrix F has been calculated as if the considered system was continuous in time. Obviously, this is not true because the considered system is time discrete. The matrix P is calculated in the following way:

$$\begin{cases} \Delta P & = (F \cdot P + P \cdot F' + Q) \cdot Ts; \\ P_{\text{new}} & = P + \Delta P; \end{cases} \tag{5.18}$$

Finally, it is possible to compute the Kalman gain and the correction to apply to the state variables.

$$K = P \cdot H' \cdot (H \cdot P \cdot H' + R)^{-1} \tag{5.19}$$

$$X = X + K \cdot (Y - Y_{\text{est}}) \tag{5.20}$$

All the proposed actions are generally always possible, even if some operations can lead to unstable results if the vehicle speed is lower than a threshold value. For this reason, in some cases while a speed lower than 5 km/h is detected by the estimation the value of the sideslip angle is forced to zero, while the one of the Yaw Rate is forced to the one of the IMU sensor output. This operation is done to avoid computational errors, without compromising the correctness of the results.

5.4 Results on Simulation

As stated in previous sections, firstly a single-track Model has been developed to estimate the sideslip angle dynamically. Firstly, a simulation on the Software Vi-Grade has been performed to verify the correctness and the validity of this model and the results have been reported in the following graph.

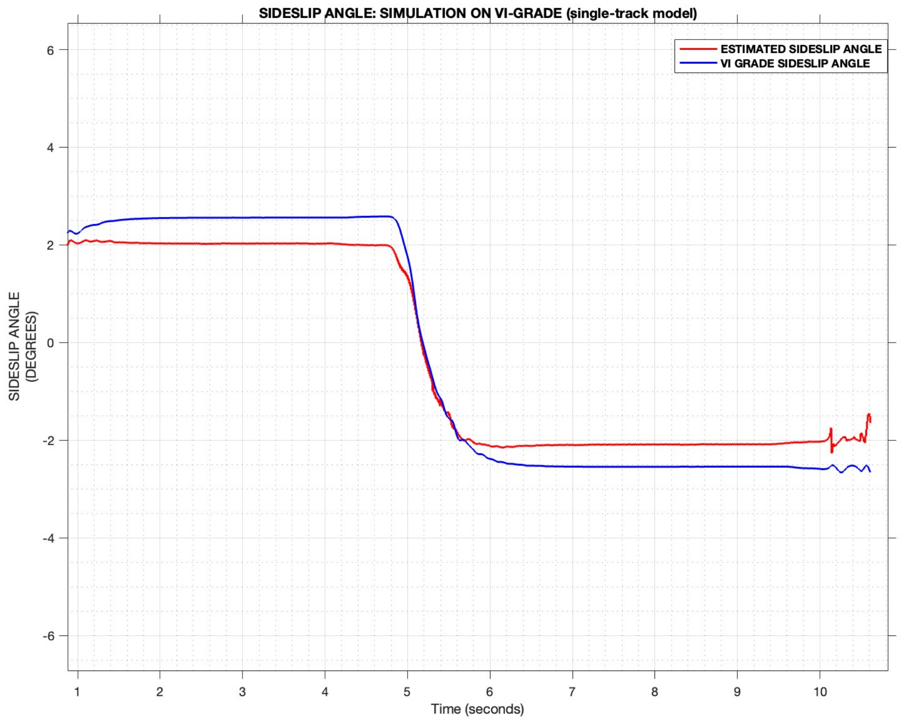

Figure 5.9: Sideslip Angle: Results on Simulation. Dynamic Filter with SingleTrack Model

It is possible to notice that a Single-Track Model is not a proper model for this kind of application. In fact, in this model the nonlinearities of the cornering stiffnesses respect to the load are not well considered as explained in the Tire Behavior section. This leads to an underestimation of the cornering stiffness of the axle, leading to an overestimation of the sideslip angle of the tire and consequently errors in the vehicle sideslip angle. For this reason, the semi-dual track model discussed in the previous subsections has been developed leading to a big improvement of the results.

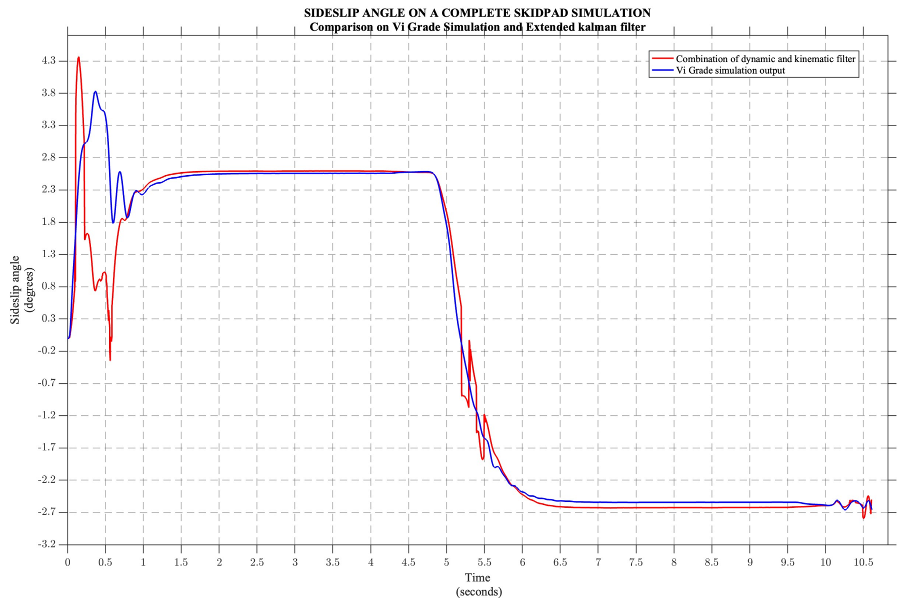

Figure 5.10: Sideslip Angle: Simulation Results with Pseudo Dual Track Model

As it is possible to notice from the graph, the steady-state error tends to zero, due to the fact that the semi-dual-track model provides a very good estimation of the tire sideslip angle α and, consequently, the vehicle sideslip angle β. In transient phases, the behavior is very good, even if there are small differences between the estimated values and the simulation output. This is true since the estimation model relies on various assumptions. The camber angle effects, roll and gravity effects, and aerodynamics effects are neglected. Moreover, there is a discrepancy between the simulation time-step of 0.001s and the frequency of the simulated sensors set at 0.01s, which results in a mismatch in the output frequency.

Additionally, the simulated sensors have an accuracy of 0.1 m/s^2, while the simulation outputs have an accuracy of 0.01 m/s^2. Furthermore, the forces exchanged by the tires match those obtained through Pacejka's formulas only after a transient phase This transient behavior can be modeled as a low-pass filter with a transfer function given by the following equation.

$$H(s) = \frac{1}{1 + \tau s} \tag{5.21}$$

with $\tau = 0.1s$

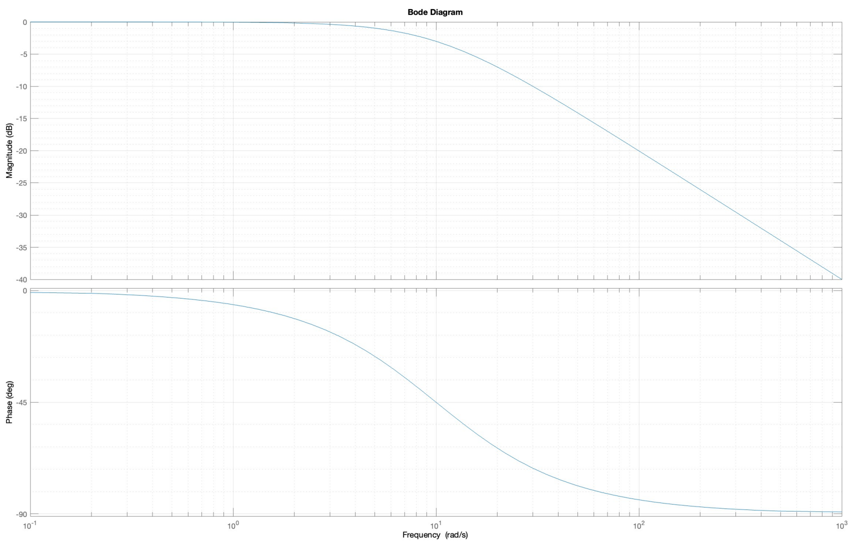

Figure 5.11: Bode Diagram of Tyre Transfer Function

Chapter 6

Combination of Kinematic and Dynamic Approach

6.1 Offline Dynamic Filter

Once the calibration of the model on simulation is concluded with enough accuracy, it is possible to adapt the proposed to real application. Before discussing about real time application, it is necessary to verify if it is possible to use the proposed method in offline application, substituting the simulated sensor with real data measured on track and compare the results obtained with real the sideslip angle directly measured by means of a Ground Speed Sensor.

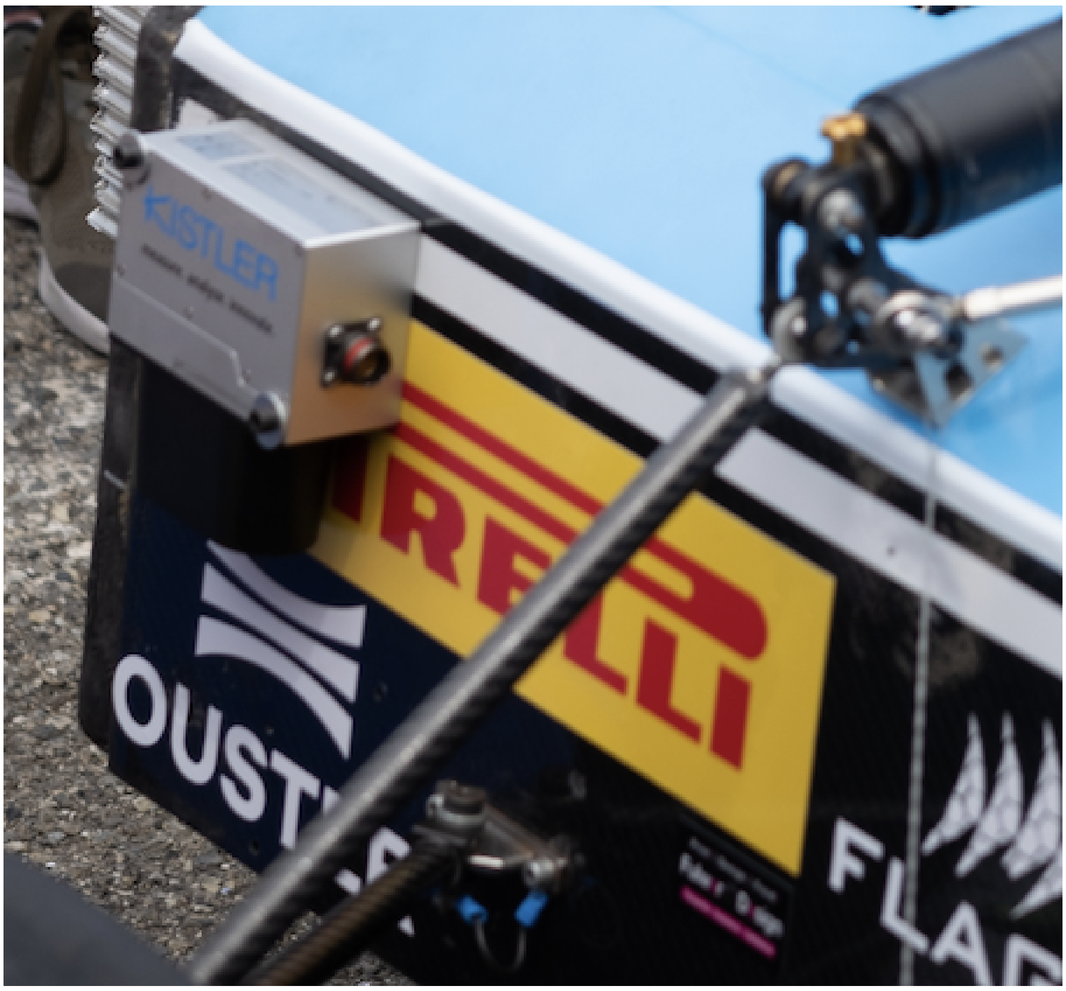

Figure 6.1: Kistler Ground Speed Sensor mounted on car

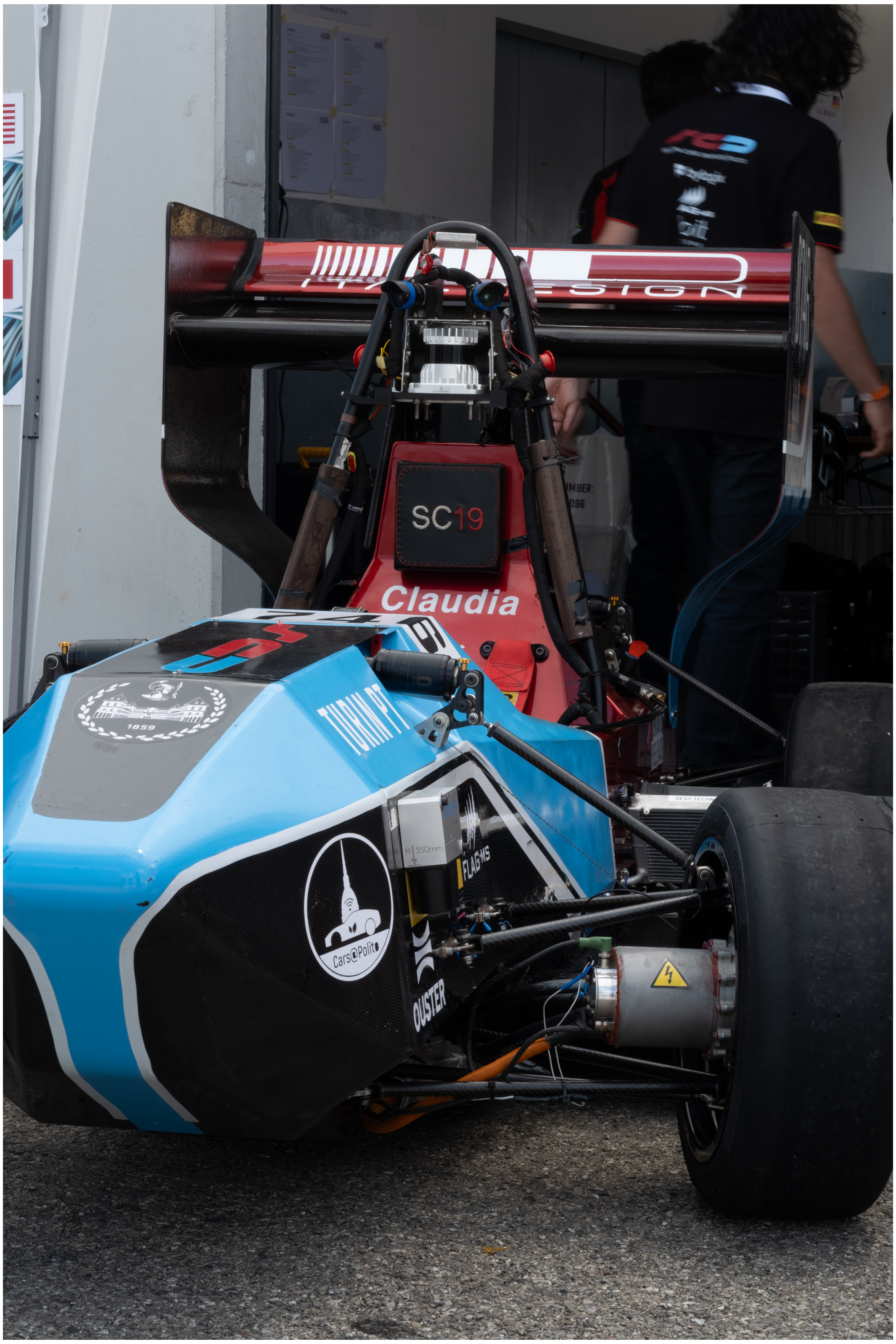

Figure 6.2: Positioning of the ground Speed Sensor on the car

A test has been done at Cerrina Track. Some cones have been positioned to create a Formula Student Driverless type circuit and multiple tests have been performed (with a driver) at speed compatible with Formula Student application (due to the presence of a driver higher speeds have been achieved without any problems). Some graph reporting the lap odometry, the measured speed and the measured sideslip angle trough the GSS are reported.

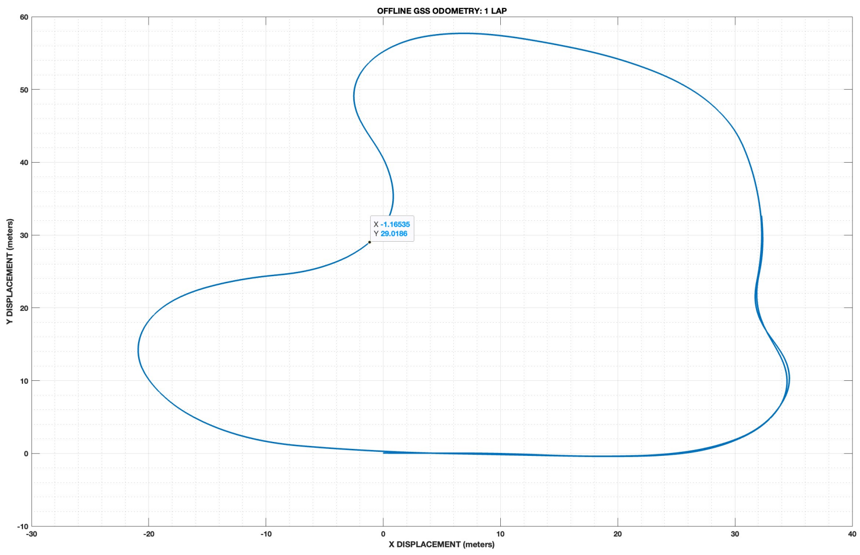

Figure 6.3: GSS: Lap Odometry

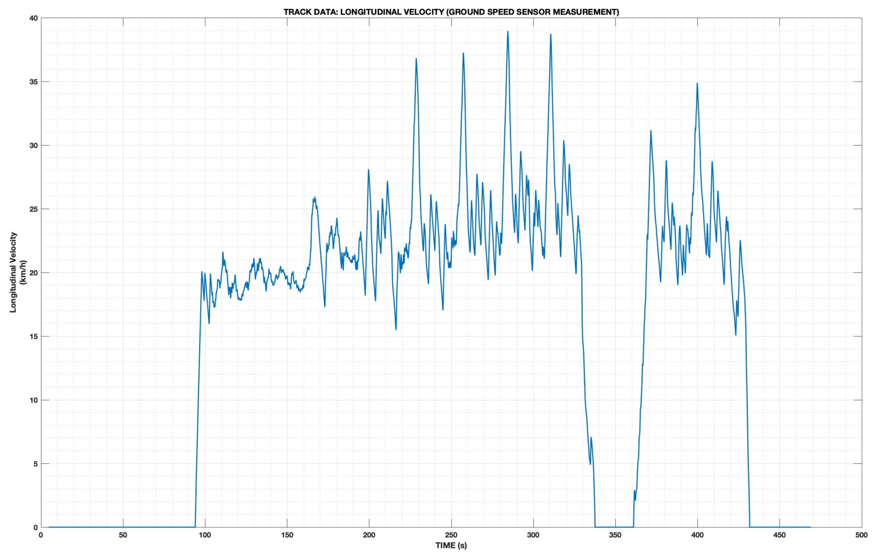

Figure 6.4: GSS: Longitudinal Velocity

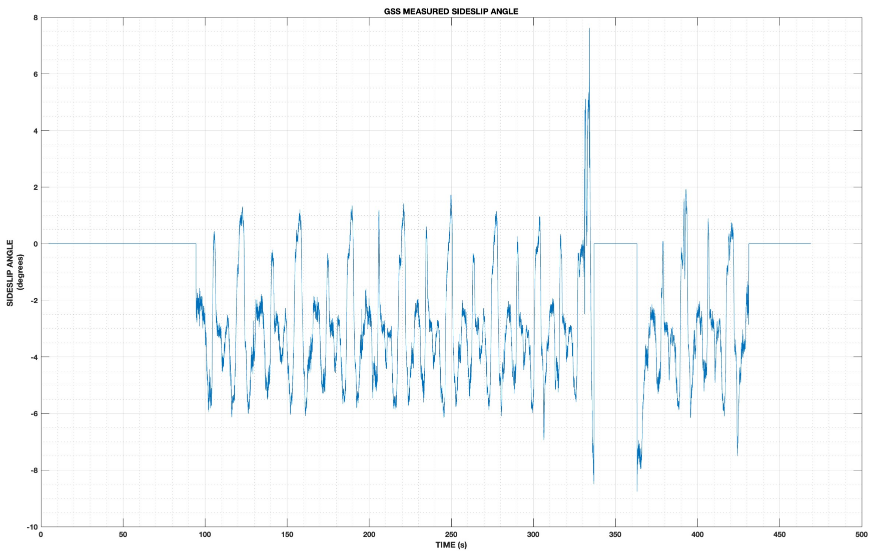

Figure 6.5: GSS: Sideslip Angle

Obviously, before inserting the sensors output in the model some considerations are necessary. These rough data are very sensible to sensors noises, vibration, positioning and ageing. For this reason, it is important to filter and remove offset and filter the outputs data.

Filtering IMU (Inertial Measurement Unit) sensor data is a common practice to enhance the quality and reliability of the measurements obtained from these sensors. Firstly, IMU

sensors can generate noisy data due to electronic interference, vibration, and other environmental factors. Filtering helps reduce this noise, resulting in smoother and more accurate measurements.
Over time, IMU sensors can experience drift, which is a gradual deviation from the true values. Filtering techniques can be used to compensate for drift and maintain accurate sensor readings.
Raw sensor data may contain rapid fluctuations or high-frequency noise that can make it challenging to interpret. Filters can smooth out these fluctuations, making the data more understandable and usable. IMUs typically combine data from accelerometers, gyroscopes, and sometimes magnetometers. Filtering helps integrate this multi-sensor data to provide more accurate information about orientation, position, and motion. IMUs may provide data at high frequencies, and some applications may not require this level of detail. Filters can be used to reduce the data rate while maintaining important information, which can be beneficial for computational efficiency.

For this reason, some filters have been applied to the sensors output. Since, a filter produces a delay phase (leading to a time delay of the information) the right compromise between goodness of filtering and phase delay should be found. One of the advantages of offline application is the fact that it is possible to filter data without introducing any delay and compare the results obtained with filtered data with the measured ones. The following filter with the following characteristics has been applied to IMU Sensor and delta position sensor.

- **PassbandFrequency**: 0.01
- **StopbandFrequency**: 0.3
- **PassbandRipple**: 0.01
- **StopbandAttenuation**: 60
- **DesignMethod**: 'ellip'

$$H(z) = \frac{0.03320z^5 - 0.06137z^4 + 0.03320z^3 + z^2 - 1.94880z + 0.95194}{0.01532z^5 + 0.01532z^4 + 0z^3 + z^2 - 0.95093z + 0} \tag{6.1}$$

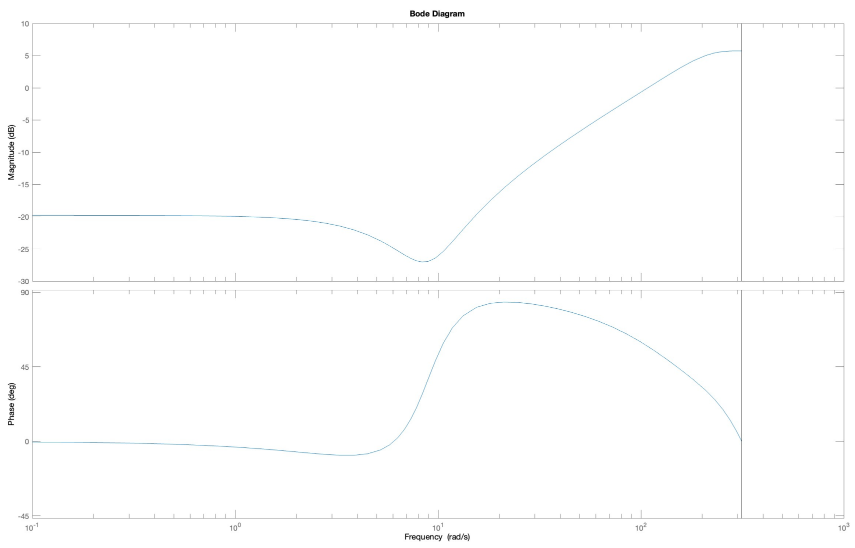

Figure 6.6: Bode Diagram Filters

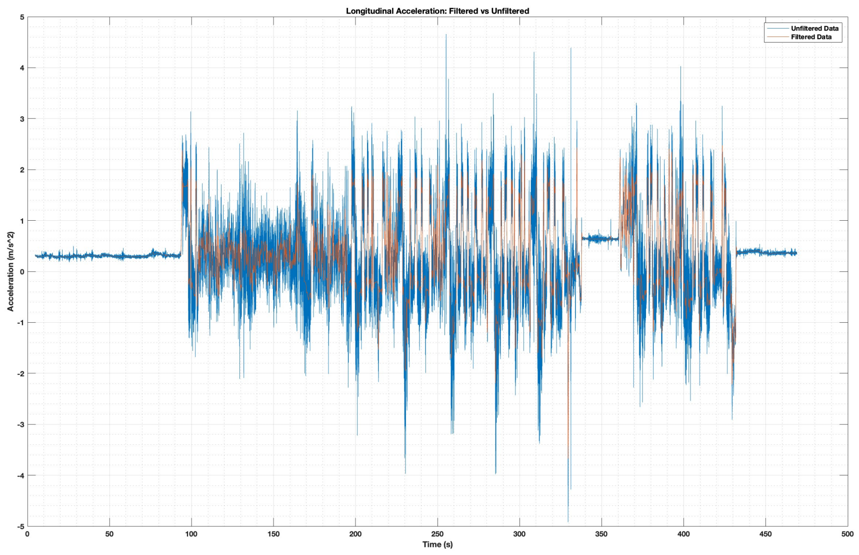

Figure 6.7: Longitudinal acceleration: Filtered vs Unfiltered Data

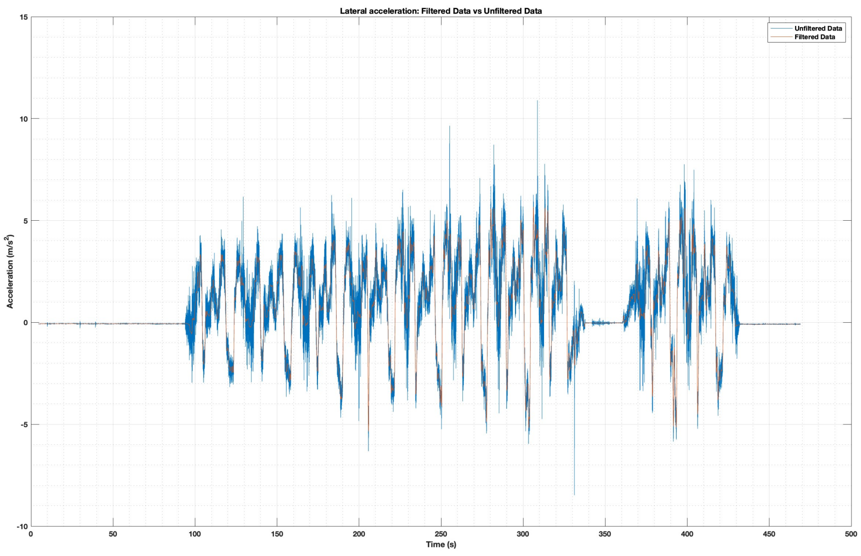

Figure 6.8: Lateral acceleration: Filtered vs Unfiltered Data

At this point, it is possible to load the data in the estimation model and obtain results.

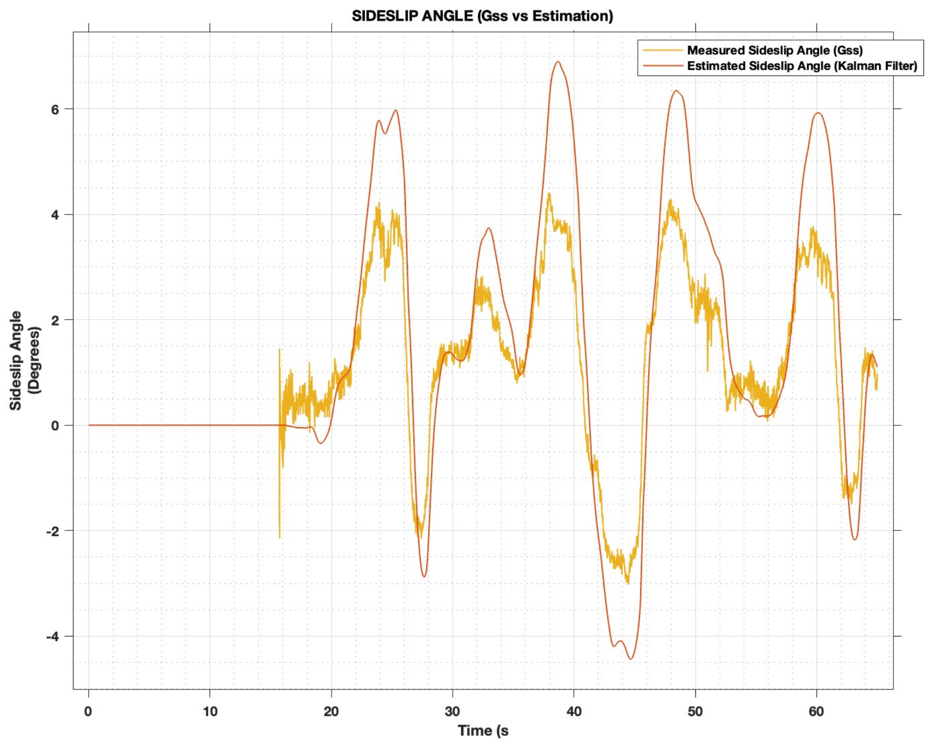

Figure 6.9: Dynamic Filter Results with Real Track Data

Table 6.1: Track Results: Sideslip Angle error: Dynamic Filter

Error Type	Value in radians	Value in degrees
Root Mean Square (RMS) Error	0.0376 rad	2.1532 deg
Mean Error	0.0080 rad	0.4587 deg
Mean Absolute Error	0.0264 rad	1.5115 deg

As it is possible to notice the dynamic filter that showed very good results in simulation does not reply its efficiency if used in real application. In fact, as stated in previous chapter one of the main disadvantages of dynamic filter respect to kinematic one is its high dependance on car parameters. Particularly, most of the parameters that in simulation can be replaced with the desired value, such as vehicle mass, vehicle inertia, tire coefficients are not known with enough accuracy and low uncertainty. For this reason, the behavior in real application can be very different respect to the one in simulation mode.

Anyway, the Kalman filter ensures the convergence of the model even with real data. Moreover, in steady state conditions and where higher forces are not exchanged the behavior of the Kalman filter is very good and very high accuracy is reached. On the other hand, in high dynamic conditions, where forces exchanged are higher a very good knowledge of the parameters is required. Also, the tire behavior is modelled accordingly to Pacejka's coefficients that are strictly dependent on tire wear, temperature, friction coefficient and road condition. For this reason, the Kalman Filter is not able in this phase to correctly estimate the sideslip angle.

Other causes of error, as already stated, can be sensors ageing, noises and drift, that even if counterbalanced by the Kalman corrections are present and unavoidable. For this reason, a better solution should be found.

6.2 Introduction to Combination

As stated in previous chapters and sections, the Dynamic Filter presents very good results in simulation (much better than the ones displayed by the Kinematic one). On the other hand, if only the derivative of the sideslip angle is considered, very interesting results come from the kinematic filter. This is due to the fact that the kinematic derivative can be calculated accordingly to simple algebraic operations, without considering any parameter and any physical law. For this reason, a way to combine the sideslip angle output of the Dynamic Filter and the kinematic sideslip angle derivative could be found.

The combination of kinematic and dynamic Kalman Filter is widely adopted in literature for the estimation of the sideslip angle of a vehicle. While almost every algorithm is based on the usage of the sideslip angle for what concerns the dynamic filter, different approaches are used for the kinematic filter. Particularly some algorithms use a combination of dynamic and kinematic filter, while others use a combination of dynamic sideslip angle and kinematic sideslip angle derivative.

Moreover, most of the algorithms use a weighted average of the two filters (the sum of the two filters weights is equal to 1), while there are present some exceptions that allow a weighted sum (with coefficients different than 1).

Here a quick description of the combination algorithms most diffused in literature is present.

In [12] the approach described involves running both the kinematic filter and dynamic filter simultaneously. The final sideslip angle estimate is obtained by taking a weighted average of the sideslip angles computed by these two filters. This process is as follows:

1. The measured lateral acceleration (a_y) is stored in a 0.1-second buffer.
2. A steady-state index is calculated, primarily based on the Root Mean Square (RMS) of the stored a_y samples.
3. The steady-state index is then used to compute weights for the kinematic contribution (w_{kin}) and dynamic contribution (w_{dyn}), with the constraint that $w_{\text{kin}} + w_{\text{dyn}} = 1$. The rationale is to use kinematic and dynamic models for transient and steady-state conditions, respectively.

Here are further details:

- If the computed RMS value of lateral acceleration (a_y) is less than $0.4\,\text{m/s}^2$ (indicating minimal variation), the steady-state index is set to 1.
- If the RMS value falls between $0.4\,\text{m/s}^2$ and $0.6\,\text{m/s}^2$, the steady-state index varies linearly from 1 to 0.

- For values greater than $0.6\,\mathrm{m/s^2}$, indicating transient conditions, the steady-state index is set to 0.
- In the range of $\pm 1\,\mathrm{m/s^2}$, the steady-state index is set to 1 to prevent fluctuations in the sideslip angle estimation during nearly straight-line conditions due to the kinematic filter. The weight w_{dyn} is 1 when the steady-state index is 1 and linearly decreases to 0.7 for a steady-state index of 0.

Additionally, this paper introduces the concept of cross-combination of common variables between the two filters. The common variables are r and v_x:

- The kinematic filter requires r as input and produces v_x as output.
- The dynamic filter needs v_x as input and produces r as output.

This novel approach, known as cross-combination, has the potential to enhance the accuracy of v_y estimation and, consequently, the sideslip angle estimation.

In [15] The proposed estimation method combines the advantages of two previous methods. It uses a weighted average to obtain the combined estimated slip angle $\hat{\beta}_{\text{comb}}$ based on the dynamic ($\hat{\beta}_{\text{din}}$) and kinematic ($\hat{\beta}_{\text{kin}}$) estimates. This method offers robustness against sensor bias errors and road bank angles at low frequencies and provides accurate dynamic changes in slip angle at higher frequencies.

The combined estimated slip angle can be expressed as:

$$\hat{\beta}_{\text{comb}}(s) = \frac{1}{\tau s + 1} \cdot \beta_{\text{din}}(s) + \frac{\tau}{\tau s + 1} \cdot \beta_{\text{kin}}(s)$$

Where τ is a filter parameter, typically set to $\tau = 1$. This formulation effectively combines the two estimates, and the method can handle situations where the vehicle exceeds its handling limits for short periods (a few seconds) quite well, as supported by experimental results.

6.3 Adopted Method and Filters

The adopted method is a freely inspired by the strategy introduced above. Particularly it combines the sideslip angle obtained by the dynamic Kalman filter and the sideslip angle derivative obtained by the kinematic Kalman filter.

For this reason, a way to combine and sum two outputs with different meaning and unit of measurement should be found. A common approach consists of filtering the two quantities with two different filters and different low pass frequency. For this reason, two transfer function (they are reported in continuous domain, but then they are converted in discrete domain) have been detected to obtain a reasonable way to sum the two quantities.

For what concerns the dynamic filter, its output already represents a sideslip angle, so it is not necessary to change the meaning of the information and for this reason a simple low pass filter is necessary.

For what concerns the kinematic filter, its output is a sideslip angle derivative, for this reason a different transfer function is necessary. The scope of this filter is the one to give an output that represents a variation of the sideslip angle (rad) and no more a derivative (rad/s).

The two transfer functions are chosen and are reported in continuous time domain. (For their adaptation in discrete time domain Matlab toolbox have been utilized, accordingly to the time step of the simulation).

Dynamic Filter:

$$T(s) = \frac{1}{1.592s + 1} \tag{6.2}$$

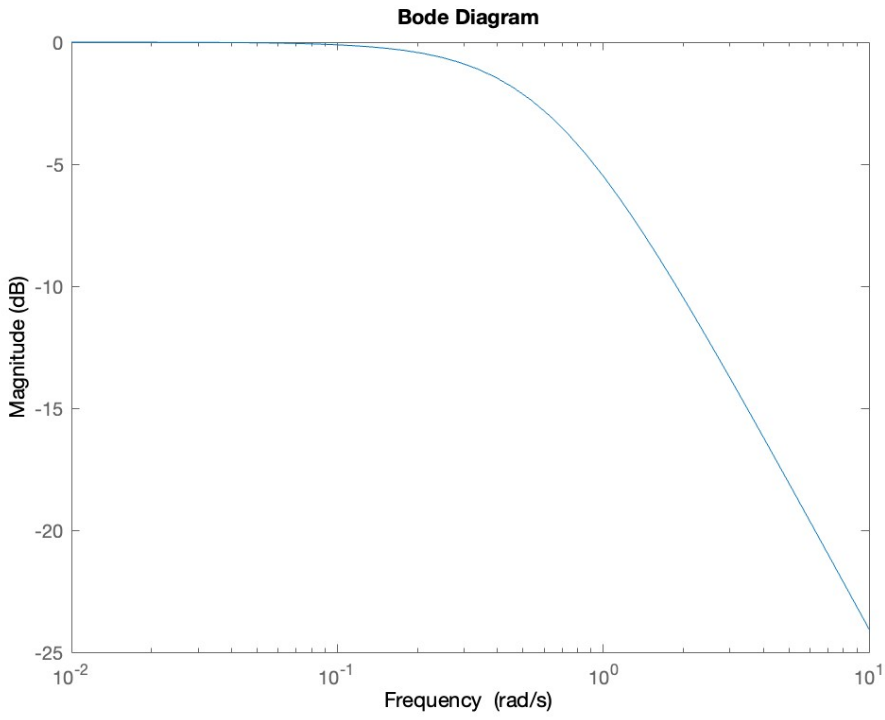

Figure 6.10: Dynamic Filter Transfer Function

Kinematic Filter:

$$T(s) = \frac{0.4775}{0.4775s + 1}$$

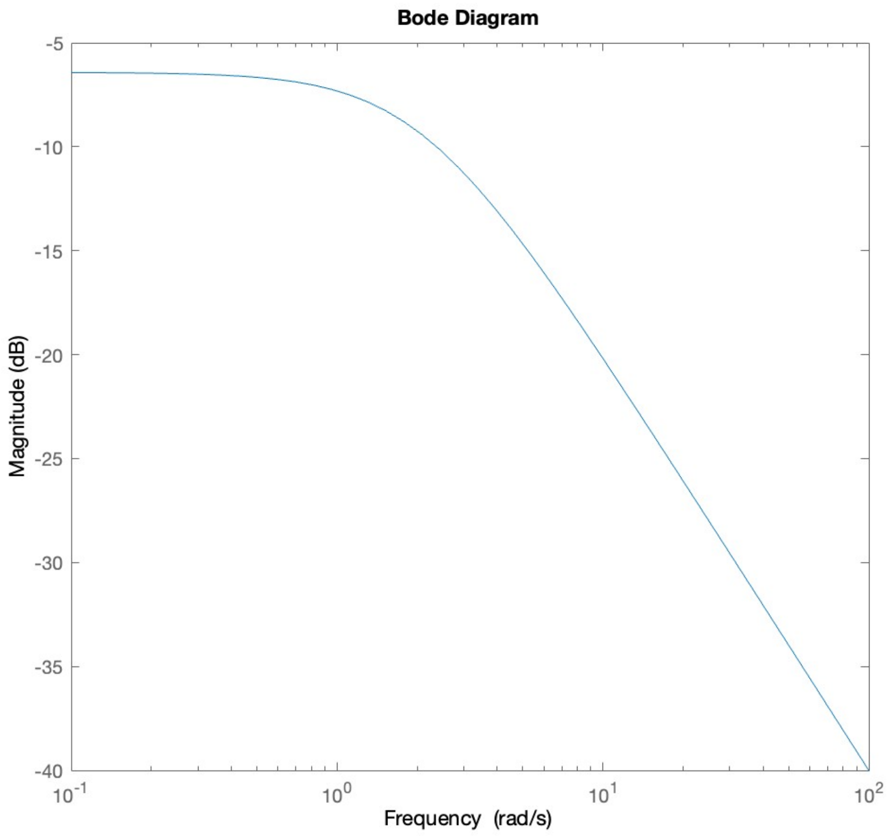

Figure 6.11: Kinematic Filter Transfer Function

At this point a way to sum the two quantities should be found, considering the conditions in which the two filters display a better behavior. As stated previously, the dynamic filter is quite robust, since it does not depend strongly on the sensor's accuracy, sensibility, and noises. For this reason, the output of this filter can be considered reliable in almost all the conditions, but not accurate in strong transient phases and for very high values of lateral acceleration.

On the other hand, the Kalman filter output (derivative of steady state angle) does not produce any kind of information in steady-state conditions and for low lateral acceleration, while it replies very well the behavior in transient phases and for big lateral accelerations.

An algorithm able to take all the advantages of the two filter should be used. Firstly, the last 10 values of the lateral acceleration are taken and the rms value is calculated. This quantity is used to calculate the Steady State Index. The Steady State Index gives information about how much the output of the sensors is varying (transient phases) and is calculated in the following way.

$$\text{Steady State Index} = \begin{cases} 1 & \text{if } \mathrm{RMS}(a_y) < 0.7 \\ 1 - \frac{\mathrm{RMS}(a_y) - 0.7}{0.3} & \text{if } 0.7 \leq \mathrm{RMS}(a_y) < 1 \\ 0 & \text{if } \mathrm{RMS}(a_y) \geq 1 \end{cases} \tag{6.4}$$

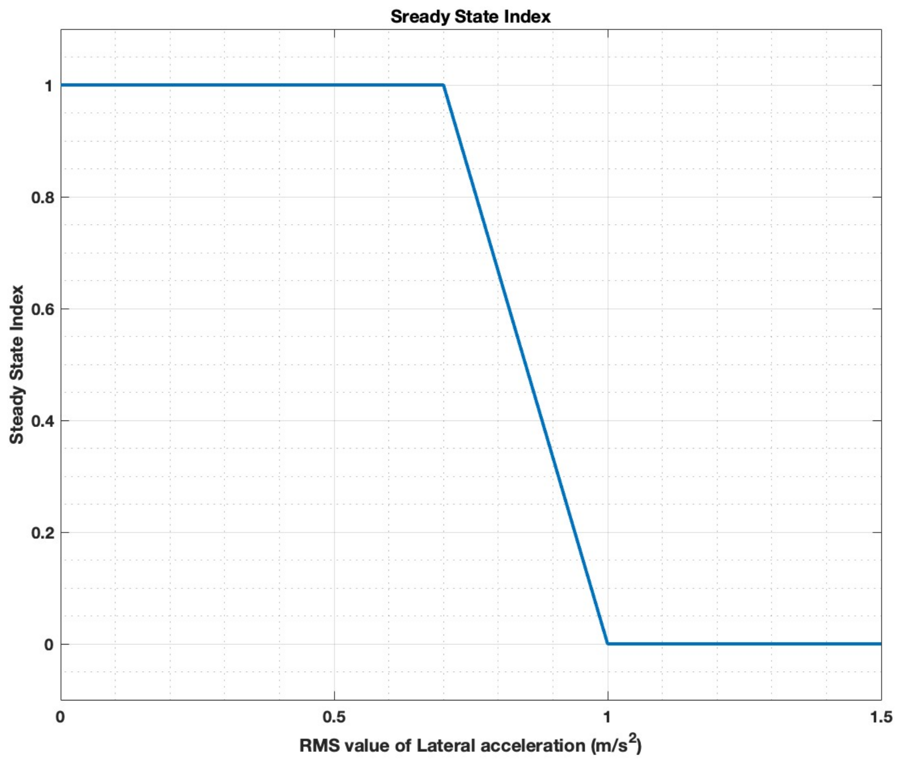

Figure 6.12: Steady State Index

At this point this information is used to calculate how much weight assign to the dynamic and kinematic Kalman filter output. The final value of the sideslip angle will be evaluated in the following way:

$$\beta = K_d \cdot \beta_{dyn} + K_k \cdot \beta_{kin} \quad (K_d + K_k = 1 \text{ e } K_d \geq 0.7) \tag{6.5}$$

The two weights strongly depend on the SSI index and are calculated in the following way.

$$\text{Coefficients} = \begin{cases} K_d = 0.7 + \mathrm{SSI} \cdot 0.3 \\ K_k = 1 - K_d \end{cases} \tag{6.6}$$

As it is possible to notice, the minimum value of the weight assigned to the dynamic filter output is equal to 0.7, since it is very reliable, while the kinematic one varies from 0 (steady state, no information) to 0.3 (very strong transient phase) since it strongly depends on sensors quality.

The choice of these values is freely inspired by other strategies adopted in literatures and have been calibrated to Formula Student Driverless application.

Chapter 7

Results

7.1 Sideslip Angle: Combination vs GSS

In this chapter the results obtained with the combination of the two filters will be introduced and discussed. In this chapter only results obtained with real track data will be discussed (in previous sections it was already stated that dynamic filter was enough for simulation).

In the following graph and table the results obtained are shown.

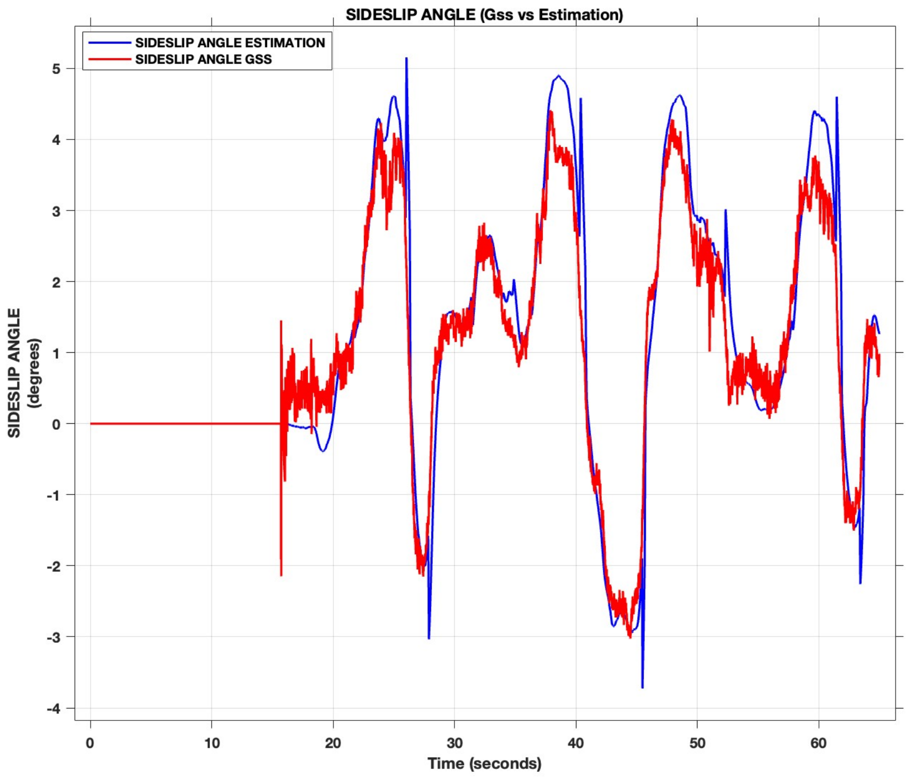

Figure 7.1: Combined Filter Results

Table 7.1: Track Results: Sideslip Angle error

Error Type	Value in radians	Value in degrees
Root Mean Square (RMS) Error	0.0086 rad	0.4918 deg
Mean Error	4.43e-4 rad	0.0052 deg
Mean Absolute Error	0.0052 rad	0.3006 deg

$$\text{Goodness of Fit} = 74.48\% \tag{7.1}$$

calculated as:

$$\text{FIT} = 100 \times \left(1 - \frac{\text{norm}(error)}{\text{norm}(Beta_{est} - \text{mean}(Beta_{est}))}\right) \tag{7.2}$$

As it is possible to notice the quality of the results is very high (very low errors have been achieved). The knowledge of the value of the sideslip angle with this accuracy is certainly very important and helpful for Formula Student Driverless Application.

The results improved very much after the application of the Combined Filter. The following graph and table shows the difference of the results obtained in both configurations.

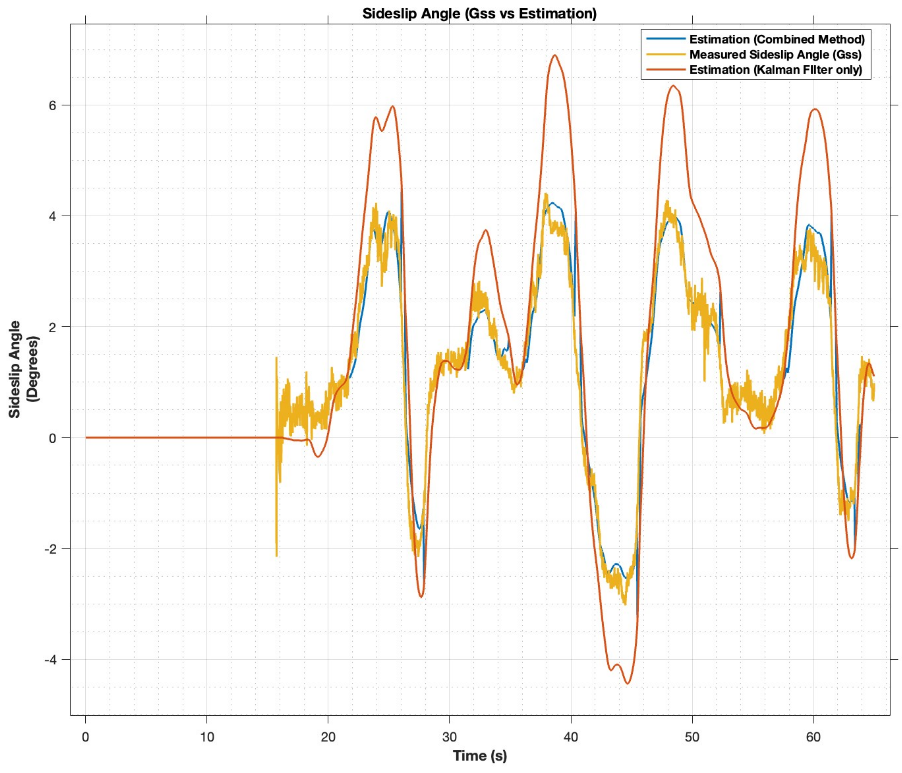

Figure 7.2: Track results: Combined vs dynamic Filter

Table 7.2: Track Results: Sideslip Angle Error. Combined vs Dynamic Filter

Error Type	**Dynamic Filter (rad)**	**Combined Filter (rad)**	**Dynamic Filter (deg)**	**Combined Filter (deg)**
Root Mean Square (RMS) Error	0.0376 rad	0.0086 rad	2.1532 deg	0.4918 deg
Mean Error	0.0080 rad	4.43e-4 rad	0.4587 deg	0.0052 deg
Mean Absolute Error	0.0264 rad	0.0052 rad	1.5115 deg	0.3006 deg

7.2 Discussion and analysis

This section delves into the discussion of the presented results.

As previously highlighted, a substantial improvement has been realized through the integration of the combined filter. This innovative approach successfully mitigates the key drawbacks associated with both the kinematic and dynamic filters.

The dynamic filter demonstrates superior performance under steady-state and low acceleration conditions. This effectiveness stems from a reduced dependence on car parameters during these phases. However, in simulation environments, this effect is not mirrored due to the absence of parameter uncertainty (each parameter in the simulation is set to the desired value). In real-world applications, where accurate knowledge of vehicle mass, inertia, and tire parameters is challenging, the quality of results is significantly compromised under such conditions.

Conversely, the kinematic filter excels in driving conditions where parameter accuracy is uncertain, relying solely on sensor quality. It proves particularly robust under higher acceleration values, where the impact of noise and offset represents only a minor percentage. Consequently, a substantial improvement in results was observed after implementing the combination of both filters.

The achieved results are undoubtedly satisfactory for Formula Student Driverless applications. However, it's crucial to emphasize their applicability primarily to the specific tests conducted (representative of nearly all events in this competition). Notably, the algorithm, especially the dynamic filter, exhibits a strong dependence on the initialization of Kalman Filter matrices. Different initial values have yielded varying results. Therefore, it is premature to consider this the final version of the Kalman Filter. Calibration, especially for a broader range of tests, remains imperative. The same applies to the calibration of the Combined Filter, necessitating attention to the Steady State Index Function and associated weights.

Chapter 8

Odometry

8.1 Offline Computation: GSS

As indicated in the Introduction, the State Estimation addresses two main objectives: the enhancement of low-level control and the refinement of track mapping through odometry.

This chapter explores the discussion of odometry derived from sensor integration. Initially, the general formula for odometry is presented, and subsequently, the results obtained from estimation and implementation for real-time applications are discussed. The following introduces the generic algorithm for Track Mapping:

$$\begin{cases} X(1) & = 0; \\ Y(1) & = 0; \\ \theta(1) & = 0; \\ \text{for } i & = 2 : \text{length}(time) \\ & \quad dt(i) = (time(i) - time(i-1)); \\ & \quad velY_{\text{gss\_ms\_cm}}(i) = velY_{\text{gss}}(i) - (yaw\_rate(i)) \cdot X_{\text{gss}}; \\ & \quad velX_{\text{gss\_ms\_cm}}(i) = velX_{\text{gss}}(i) + (yaw\_rate(i)) \cdot Y_{\text{gss}}; \\ & \quad \theta(i) = \theta(i-1) + ((yaw\_rate(i)) \cdot dt(i)); \\ & \quad X(i) = X(i-1) + (\cos(\theta(i)) \cdot velX_{\text{gss\_ms\_cm}}(i) \cdot dt(i) - \sin(\theta(i)) \cdot velY_{\text{gss\_ms\_cm}}(i) \cdot dt(i)); \\ & \quad Y(i) = Y(i-1) + (\cos(\theta(i)) \cdot velY_{\text{gss\_ms\_cm}}(i) \cdot dt(i) + \sin(\theta(i)) \cdot velX_{\text{gss\_ms\_cm}}(i) \cdot dt(i)); \\ \text{end} & \end{cases} \tag{8.1}$$

where X_{gss} and Y_{gss} represent the coordinates of the position of the Ground Speed Sensor (front wing) respect to the center of mass of the vehicle. All the velocities are in this way referred to the center of mass of the vehicle.

In offline application the computation of the track mapping and odometry is very easy, since the Ground Speed Sensor gives as output not only the value of the velocities, but also the exact time at which they are referred. Consequently, track mapping is nearly flawless, and the results are presented for a single lap:

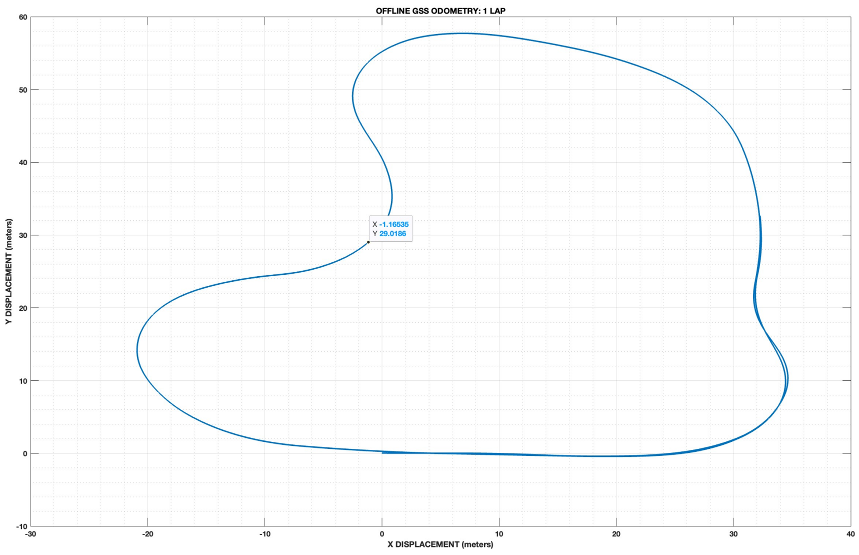

Figure 8.1: GSS Odometry: 1 lap

Moreover, as confirmation of the validity of this model the graph reporting consecutive laps is reported. As it is possible to notice the convergence of the results is almost perfect and the end of each lap coincides perfectly with the start of a new lap.

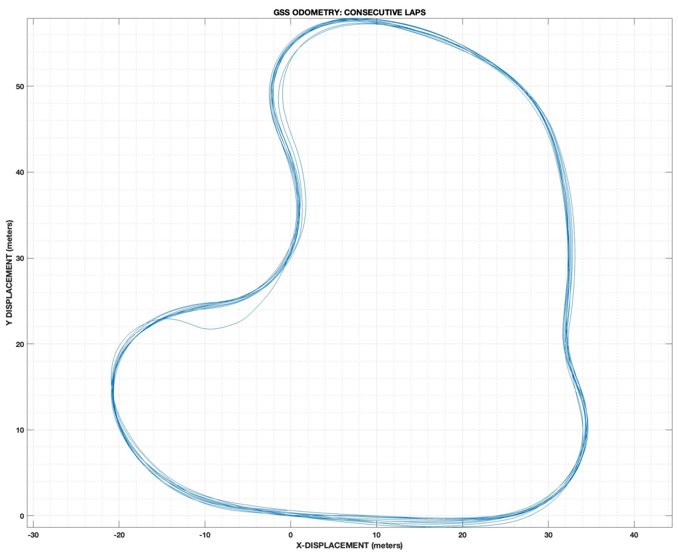

Figure 8.2: GSS Odometry: Consecutive Laps

The inclusion of a Ground Speed Sensor results in highly accurate outcomes. The subsequent section explores the scenario where a perfect estimation, characterized by results perfectly aligned with those of the Ground Speed Sensor, is present.

8.2 Online Computation: GSS

This section delves into the outcomes achieved through the implementation of a hypothetical perfect estimation. The primary distinction from the previous scenario lies in the fact that the estimation outputs are not accessible at the same frequency as the Ground Speed Sensor but only at specified intervals determined by the simulation time step (in the team's application, it is set to 0.01 seconds). Consequently, a considerable amount of information is lost, and the earlier formula can only be applied at certain higher frequencies. The ensuing results are now presented:

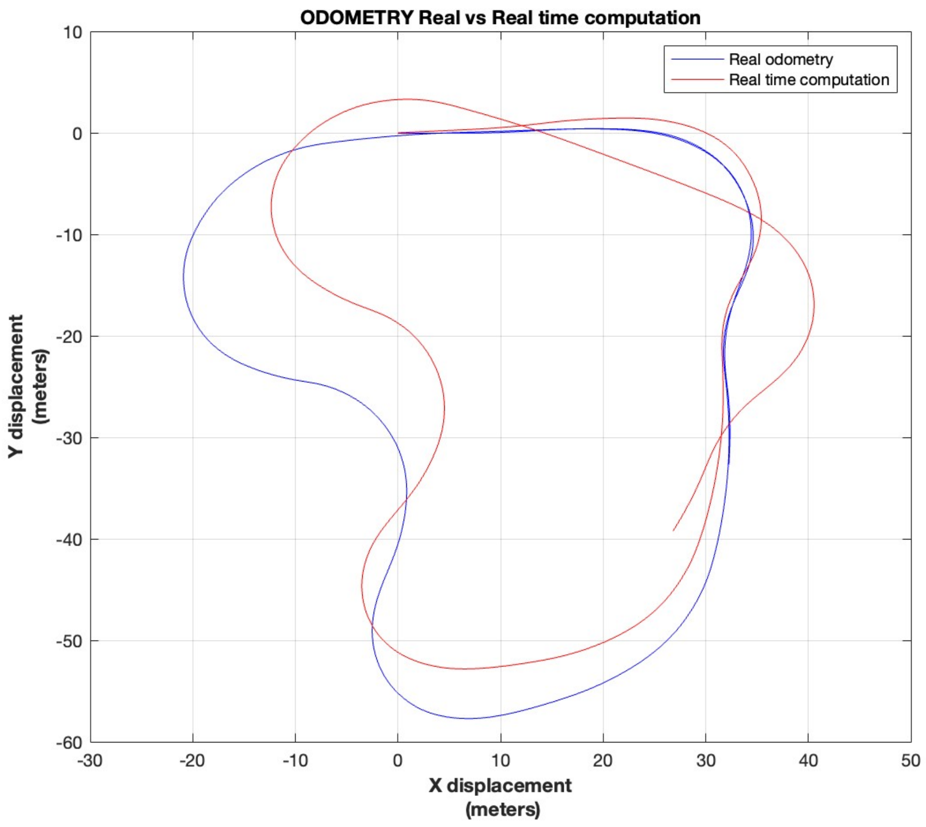

Figure 8.3: Odometry: Offline vs Real Time

As evident from the outcomes, the results obtained are significantly inferior to the previous ones. This is attributed to the loss of a substantial amount of information, making it impossible to achieve perfect track mapping with a model having this timestep.

8.3 Online Computation: Estimation

As stated all the simulink model is able to run in real time with a time step of 0.005s, thanks to the dSpace ECU.

DSpace is a platform used for developing and testing real-time control systems. When running a Simulink model on ECU (Electronic Control Units) several steps should be followed.

Firstly, the Simulink model is created, using various blocks to represent the system and the controller. Once the model is ready, it needs to be configured to work with the DSpace hardware.

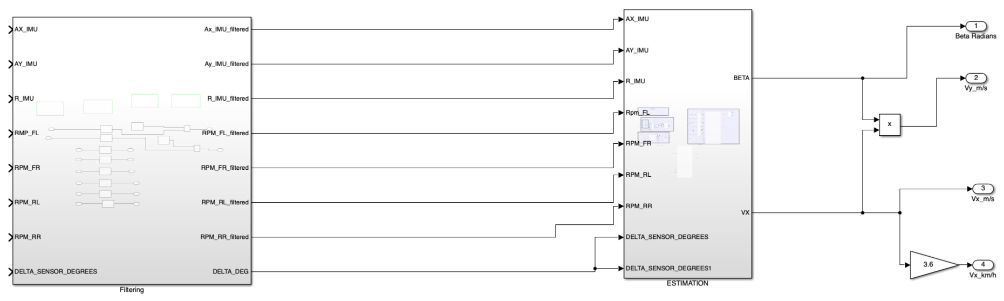

Figure 8.4: Simulink Model Subsystem: Visualization of main blocks: Filtering and Estimation

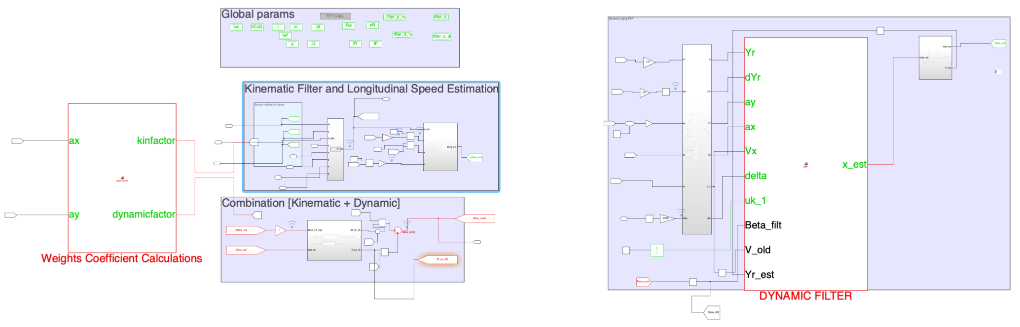
Global params
Kinematic Filter and Longitudinal Speed Estimation
Combination [Kinematic + Dynamic]
ax
ay
kinfactor
dynamicfactor
Weights Coefficient Calculations
Yr
dYr
ay
ax
Vx
delta
uk_1
Beta_filt
V_old
Yr_est
x_est
DYNAMIC FILTER

Figure 8.5: Simulink Model Estimation Subsystem: Visualization of main blocks

DSpace provides an integrated environment for this configuration. The Simulink model is connected to the DSpace platform using dedicated connectors or specific software modules provided by DSpace. Next, the Simulink model is translated into C code using built-in code generation tools within Simulink. This generated code is then compiled for the specific hardware used in DSpace.

Once the code is compiled, it's loaded onto the DSpace ECU, a process commonly known as flashing. This step allows the DSpace platform to execute the control algorithm in real-time.

DSpace's specialized hardware and software interfaces facilitate the interaction with

ECUs, enabling engineers to run Simulink models in real-time. This approach allows for thorough development and testing of control systems before their actual deployment, ensuring their effectiveness and reliability in real-world scenarios.

In this section the influence of the estimation, respect to the usage of real data coming from the ground speed sensor is discussed. As stated in previous section a big worsening of the results is caused by the fact that a lot of information is lost due to the discretization of time simulation. For this reason the usage of real data should create a bigger worsening. The effects respect to previous case and real data are now analyzed.

Firstly, the results obtained with the estimation are compared with the ones obtained through the online estimation.

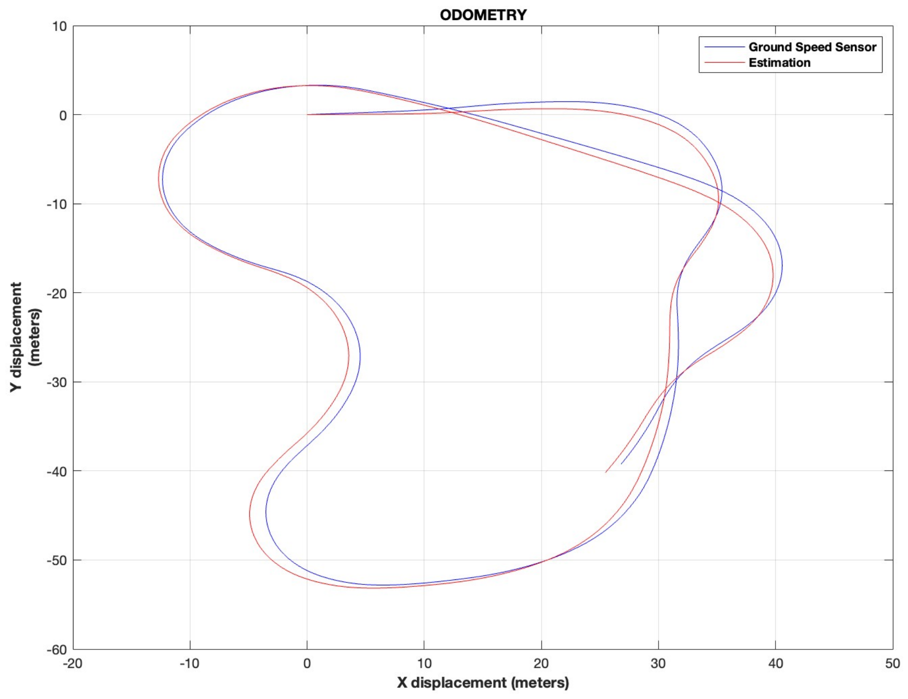

Figure 8.6: Odometry: Online Real Data vs Estimation

As evident, the quality of the results remains consistent, and the two curves almost coincide. This is attributed to the highly accurate estimation of lateral velocity, achieved through the combination of the two filters. Additionally, the estimated value of longitudinal velocity is also noteworthy, resulting in particularly favorable outcomes compared to the previous curve.

This table confirms the goodness of the estimation. In fact the difference between the results obtained using real Data coming from Ground Speed Sensor and the ones obtained Estimation Data is very similar. The accuracy is higher than 90% and allows the almost perfect estimation of the real position of the car from a theoretical point of view. The errors of few meters (1m the RMS value and 1.73 m the Max Value) are absolutely compatible with the targets of the team about Localization and Mapping. It is remarkable to remind

Table 8.1: Odometry: Errors between estimation and GSS

Variable	Quantity	Unit of Measurement
Distance: RMS value	1.0084	meters
Distance: Mean value	0.7820	meters
Distance: Max Value	1.7337	meters
Distance (x-direction): RMS Value	0.7530	meters
Distance (x-direction): Mean Value	0.5110	meters
Distance (x-direction): Max Value	1.6258	meters
Distance (y-direction): RMS Value	0.6707	meters
Distance (y-direction): Mean Value	0.4339	meters
Distance (y-direction): Max Value	1.2943	meters

that in Simultaneous Localization and Mapping, the car is able at each lap to localize itself and both recreate the map of the lap, improving errors done in previous laps.

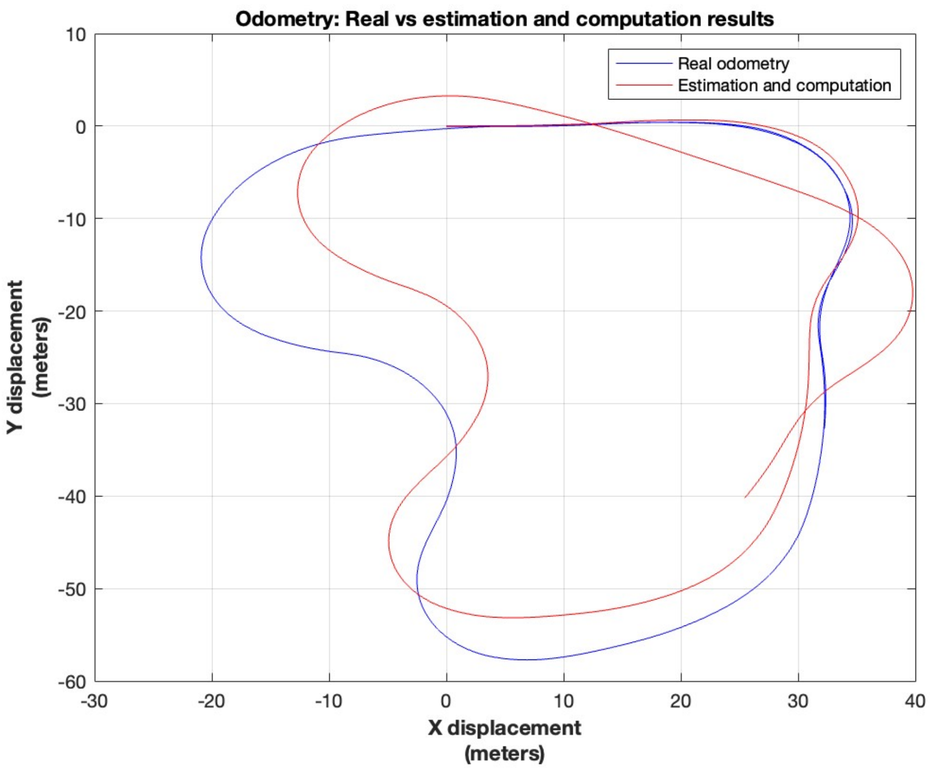

Figure 8.7: Odometry: Offine Real Data vs Estimation

This graph reports the errors obtained with the real data. As expected the errors are not so low. One big problem may arised ue to the fact that the start and the end of a lap never coincide, leading to an apparent impossibility for algorithm to detect the end of the lap. It is important to remember that there are no other flags that can help the detection of the end of a lap and that the car should be able to stop itself after the requested number of laps. This problem is amplified due to the fact that the pendence of the two curves does not coincide. Anyway thanks to the SLAM algorithm, introduced in previous chapters, with lower and lower errors it is possible to easily detect the end of a lap and the start of a new one. At the actual State the car is anyway able to stop itself at the end of a lap with accuracy of few meters. Anyway bigger improvement can be achieved with the reduction of the simulation time-step.

Chapter 9

Conclusions and Next Steps

In conclusion of this work, significant advancements have been made in the field of State Estimation for the "Squadra Corse Driverless Polito" team. The validation of previous efforts on longitudinal velocity, coupled with the successful development of an effective model for lateral velocity estimation, marks a substantial improvement.

However, as highlighted in preceding chapters, the algorithm requires calibration for a broader range of applications. Furthermore, addressing the accuracy of car parameters, such as the estimation of vehicle mass and inertia, is crucial for enhancing results. Considerable strides can be made by integrating the State Estimation Output with data obtained from the Custome Ground Speed Sensor. Rigorous testing is imperative to fine-tune an algorithm capable of seamlessly combining both sources.

Notwithstanding, the implementation of State Estimation has yielded excellent outcomes, particularly in terms of facilitating the implementation of low-level control mechanisms such as Torque Vectoring and Traction Control.

Turning our attention to odometry and track mapping, nuanced conclusions emerge. It is evident that the primary issue with inaccurate odometry stems not from faulty estimation but rather from the utilization of excessively high discretization time during simulation. Although the current work lays a foundation, especially for improving track mapping, additional enhancements are warranted. Existing literature suggests various methods for implementing odometry to augment track mapping. A model that integrates GPS position measurements with yaw rate data and odometry holds promise.

Appendix

Measurement Matrix H

$$H(1,1) = 0 \tag{9.1}$$

$$H(1,2) = 1 \tag{9.2}$$

H(2,1) =

$$\frac{C_y \cdot D_{y_f} \cdot \cos(C_y \cdot \arctan(E_{y_f} \cdot \arctan(B_{y_f} \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}) - B_{y_f} \cdot (E_{y_f} - 1) \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}))) \cdot (B_{y_f} \cdot (E_{y_f} - 1) - \frac{B_{y_f} \cdot E_{y_f}}{B_{y_f}^2 \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x})^2 + 1})}{(E_{y_f} \cdot \arctan(B_{y_f} \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x})) - B_{y_f} \cdot (E_{y_f} - 1) \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}))^2 + 1} \tag{9.3}$$

H(2,2)=

$$\frac{C_y \cdot D_{y_f} \cdot \cos(C_y \cdot \arctan(E_{y_f} \cdot \arctan(B_{y_f} \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}) - B_{y_f} \cdot (E_{y_f} - 1) \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}))) \cdot \left(\frac{B_{y_f} \cdot l_F \cdot (E_{y_f} - 1)}{V_x} - \frac{B_{y_f} \cdot E_{y_f} \cdot l_F}{V_x \cdot (B_{y_f}^2 \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x})^2 + 1)}\right)}{(E_{y_f} \cdot \arctan(B_{y_f} \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x})) - B_{y_f} \cdot (E_{y_f} - 1) \cdot (-\delta + \frac{Y_r \cdot l_F}{V_x}))^2 + 1} \tag{9.4}$$

H(3,1)=

$$\frac{C_y \cdot D_{y_r} \cdot \cos(C_y \cdot \arctan(E_{y_r} \cdot \arctan(B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) - B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) \cdot (E_{y_r} - 1))) \cdot \left(\frac{B_{y_r} \cdot (E_{y_r} - 1)}{V_x} - \frac{B_{y_r} \cdot E_{y_r}}{B_{y_r}^2 \cdot (-\frac{Y_r \cdot l_R}{V_x})^2 + 1}\right)}{(E_{y_r} \cdot \arctan(B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x})) - B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) \cdot (E_{y_r} - 1))^2 + 1} \tag{9.5}$$

H(3,2)=

$$-\frac{C_y \cdot D_{y_r} \cdot \cos(C_y \cdot \arctan(E_{y_r} \cdot \arctan(B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) - B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) \cdot (E_{y_r} - 1))) \cdot \left(\frac{B_{y_r} \cdot l_R \cdot (E_{y_r} - 1)}{V_x} - \frac{B_{y_r} \cdot E_{y_r} \cdot l_R}{V_x \cdot (B_{y_r}^2 \cdot (-\frac{Y_r \cdot l_R}{V_x})^2 + 1)}\right)}{(E_{y_r} \cdot \arctan(B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x})) - B_{y_r} \cdot (-\frac{Y_r \cdot l_R}{V_x}) \cdot (E_{y_r} - 1))^2 + 1} \tag{9.6}$$

Jacobian Matrix

F(1,1)=

$$-\left(\frac{Cy\cdot Dy_r\cdot\cos(Cy\cdot\arctan(Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx})-By_r\cdot(-\frac{Yr\cdot lR}{Vx})\cdot(Ey_r-1)))\cdot\left(\frac{By_r\cdot(Ey_r-1)}{Vx}-\frac{By_r\cdot Ey_r}{By_r^2\cdot(-\frac{Yr\cdot lR}{Vx})^2+1}\right)}{(Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx}))-By_r\cdot(-\frac{Yr\cdot lR}{Vx})\cdot(Ey_r-1))^2+1}\right)$$
$$+\left(\frac{Cy\cdot Dy_f\cdot\cos(Cy\cdot\arctan(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx})-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))\cdot\left(\frac{By_f\cdot(Ey_f-1)}{Vx}-\frac{By_f\cdot Ey_f}{By_f^2\cdot(-\delta+\frac{Yr\cdot lF}{Vx})^2+1}\right)}{(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))^2+1}\right) \tag{9.7}$$

F(1,2)=

$$-\left(\frac{Cy\cdot Dy_f\cdot\cos(Cy\cdot\arctan(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Y_r\cdot lF}{Vx})-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Y_r\cdot lF}{Vx}))\cdot\left(\frac{By_f\cdot lF\cdot(Ey_f-1)}{Vx}-\frac{By_f\cdot Ey_f\cdot lF}{Vx\cdot(By_f^2\cdot(-\delta+\frac{Y_r\cdot lF}{Vx})^2+1)}\right)}{(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Y_r\cdot lF}{Vx}))-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Y_r\cdot lF}{Vx}))^2+1}\right)$$
$$-\left(\frac{Cy\cdot Dy_r\cdot\cos(Cy\cdot\arctan(Ey_r\cdot\arctan(By_r\cdot(-\frac{Y_r\cdot lR}{Vx})-By_r\cdot(-\frac{Y_r\cdot lR}{Vx})\cdot(Ey_r-1)))\cdot\left(\frac{By_r\cdot lR\cdot(Ey_r-1)}{Vx}-\frac{By_r\cdot Ey_r\cdot lR}{Vx\cdot(By_r^2\cdot(-\frac{Y_r\cdot lR}{Vx})^2+1)}\right)}{(Ey_r\cdot\arctan(By_r\cdot(-\frac{Y_r\cdot lR}{Vx}))-By_r\cdot(-\frac{Y_r\cdot lR}{Vx})\cdot(Ey_r-1))^2+1}\right)-1 \tag{9.8}$$

F(2,1)=

$$\frac{Cy\cdot Dy_r\cdot lR\cdot\cos(Cy\cdot\arctan(Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx})-By_r\cdot(-\frac{Yr\cdot lR}{Vx})\cdot(Ey_r-1)))\cdot\left(\frac{By_r\cdot(Ey_r-1)}{Vx}-\frac{By_r\cdot Ey_r}{By_r^2\cdot(-\frac{Yr\cdot lR}{Vx})^2+1}\right)}{Jz\cdot((Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx}))-By_r\cdot(-\frac{Yr\cdot lR}{Vx})\cdot(Ey_r-1))^2+1)}$$
$$-\frac{Cy\cdot Dy_f\cdot lF\cdot\cos(Cy\cdot\arctan(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx})-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))\cdot\left(\frac{By_f\cdot(Ey_f-1)}{Vx}-\frac{By_f\cdot Ey_f}{By_f^2\cdot(-\delta+\frac{Yr\cdot lF}{Vx})^2+1}\right)}{Jz\cdot((Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))^2+1)} \tag{9.9}$$

F(2,2)=

$$-\left(\frac{Cy\cdot Dy_f\cdot lF\cdot\cos(Cy\cdot\arctan(Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx})-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))\cdot\left(\frac{By_f\cdot lF\cdot(Ey_f-1)}{Vx}-\frac{By_f\cdot Ey_f\cdot lF}{Vx\cdot(By_f^2\cdot(-\delta+\frac{Yr\cdot lF}{Vx})^2+1)}\right)}{Jz\cdot((Ey_f\cdot\arctan(By_f\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))-By_f\cdot(Ey_f-1)\cdot(-\delta+\frac{Yr\cdot lF}{Vx}))^2+1}\right)$$
$$-\left(\frac{Cy\cdot Dy_r\cdot lR\cdot\cos(Cy\cdot\arctan(Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx})-By_r\cdot(-\frac{Yr\cdot lR}{Vx})\cdot(Ey_r-1)))\cdot\left(\frac{By_r\cdot lR\cdot(Ey_r-1)}{Vx}-\frac{By_r\cdot Ey_r\cdot lR}{Vx\cdot(By_r^2\cdot(-\frac{Yr\cdot lR}{Vx})^2+1)}\right)}{Jz\cdot((Ey_r\cdot\arctan(By_r\cdot(-\frac{Yr\cdot lR}{Vx})-By_r\cdot(Ey_r-1)\cdot(-\frac{Yr\cdot lR}{Vx}))^2+1}\right) \tag{9.10}$$

Bibliography

[1] Society of Automotive Engineers. *Formula Sae Website*. s.d. URL: https://www.fsaeonline.com/.

[2] Squadra Corse Driverless. *Squadra Corse Driverless Website*. s.d. URL: https://squadracorsedriverless.com//.

[3] Society of Automotive Engineers. *Formula ATA Website*. s.d. URL: https://www.formula-ata.it/.

[4] X. Shen et al. "Fast joint compatibility branch and bound for feature cloud matching". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 1757–1764.

[5] Shuran Zheng et al. "Simultaneous Localization and Mapping (SLAM) for Autonomous Driving: Concept and Analysis". In: *Remote Sensing* 15.4 (2023), p. 1156. DOI: 10.3390/rs15041156. URL: https://doi.org/10.3390/rs15041156.

[6] Mathworks. *Slam Algorithm*. URL: https://uk.mathworks.com/discovery/slam.html.

[7] Matteo Frosi and Matteo Matteucci. "ART-SLAM: Accurate Real-Time 6DoF LiDAR SLAM". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 2692–2699. DOI: 10.1109/LRA.2022.3144795.

[8] Akihiro Kato and Tomi H. Kinnunen. "Statistical Regression Models for Noise Robust F0 Estimation Using Recurrent Deep Neural Networks". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.12 (2019), pp. 2336–2349. DOI: 10.1109/TASLP.2019.2945489.

[9] Hideaki Sasaki and Takatoshi Nishimaki. "A SideSlip Angle Estimation Using Neural Network for a Wheeled Vehicle". In: *SAE Technical Paper*. 2000. DOI: 10.4271/2000-01-0695.

[10] Angelo Bonfitto et al. "Combined regression and classification artificial neural networks for sideslip angle estimation and road condition identification". In: *Vehicle system dynamics* 58.11 (2020), pp. 1766–1787.

[11] Martino et al. "Real-time estimation of the vehicle sideslip angle through regression based on principal component analysis and neural networks". In: *2017 IEEE International Systems Engineering Symposium (ISSE)*. Oct. 2017. DOI: 10.1109/SysEng.2017.8088274.

[12] Daniel Chindamo, Basilio Lenzo, and Marco Gadola. "On the Vehicle Sideslip Angle Estimation: A Literature Review of Methods, Models, and Innovations". In: *Appl. Sci.* 8.3 (2018), p. 355. DOI: 10.3390/app8030355.

[13] D. Selmanaj et al. "Vehicle Sideslip Estimation: A Kinematic-Based Approach". In: *Control Engineering Practice* 67 (2017), pp. 1–12.

[14] A. Dardanelli et al. “Model-Based Kalman Filtering Approaches for Frequency Tracking”. In: *IFAC Proceedings Volumes* 43.10 (2010). 10th IFAC Workshop on the Adaptation and Learning in Control and Signal Processing, pp. 37–42. ISSN: 1474-6670. DOI: https://doi.org/10.3182/20100826-3-TR-4015.00010. URL: https://www.sciencedirect.com/science/article/pii/S1474667015323363.

[15] Murali R Rajamani. “Data-based techniques to improve state estimation in model predictive control”. PhD thesis. Citeseer, 2007.

[16] Vi-grade. *Vi-grade car real time*. s.d. URL: https://www.vi-grade.com/.

[17] Bryan McKenzieSousso KelouwaniSousso KelouwaniMarc-André GaudreauMarc-André Gaudreau. “Toward Synthetic Data Generation to Enhance Skidding Detection in Winter Conditions”. In: *World Electric Vehicle Journal* (2022). DOI: 10.3390/wevj13120231.

[18] *Kistler Correvit SF Motion*. s.d. URL: https://www.kistler.com/CA/en/cp/non-contact-optical-sensors-correvit-sf-motion-2059a/P0000168.

[19] Greg Welch and Gary Bishop. *An Introduction to the Kalman Filter*. Tech. rep. TR 95-041. Chapel Hill, NC 27599-3175: Department of Computer Science, University of North Carolina at Chapel Hill, July 2006.

[20] Dan Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley & Sons, 2006.

[21] Nenad Kranjcevic, Kristian Gruicic, and Marko Jokic. “Comparison of a 10 DOF quarter vehicle model with the FE analysis of tire impact against a road obstacle”. In: *Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb* ().

[22] Yucheng Liu. “Constructing Equations of Motion for A Vehicle Rigid Body Model”. In: *SAE International Journal of Passenger Cars - Mechanical Systems* 1.1 (2008). DOI: 10.4271/2008-01-2751.

[23] Pierdomenico Ruggieri. “Studio del modello CDTire per analisi Handling e Ride Comfort di un veicolo in relazione ai parametri geometrici e operativi degli pneumatici”. MA thesis. Politecnico di Torino, Dicembre 2018.

[24] Hans B. Pacejka. *Tire and Vehicle Dynamics*. Third. The brush model that represents the contact patch featuring horizontal tread element compliance and partial sliding. PublisherName, 2012. Chap. ChapterNumber.

[25] Hans B. Pacejka. *Tire and Vehicle Dynamics*. Third. The Magic Formula tire model to describe the nonlinear slip force and moment properties. PublisherName, 2012. Chap. ChapterNumber.

[26] H. B. PACEJKA and I. J. M. BESSELINK. “Magic Formula Tyre Model with Transient Properties”. In: *Vehicle System Dynamics* 27.sup001 (1997), pp. 234–249. DOI: 10.1080/00423119708969658. eprint: https://doi.org/10.1080/00423119708969658. URL: https://doi.org/10.1080/00423119708969658.