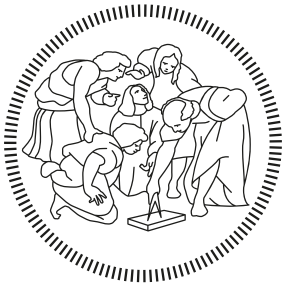
Politecnico di Milano
Master of Science in Mechanical Engineering



Sideslip estimation using a LFT-based estimator with commercial positioning and inertial sensors

Supervisor: Prof. Luca Bascetta
Assistant Supervisor: Prof. Gianni Ferretti

Graduation thesis by
Alessandro Ros
ID 858038

Academic Year 2016 - 2017

Abstract

This work focuses on the development and implementation of multiple techniques to estimate a vehicle slip angle, the offset between the tires direction and the real direction followed during a maneuver, caused by non-perfect adherence between the tires and the road. This quantity, together with the vehicle input commands, is enough to estimate and predict the lateral vehicle dynamics, allowing the introduction of advanced safety and control features. Slip angle measurement is traditionally performed only in critical situations, by using dedicated optical or inertial sensors. The spread of low cost, MEMS-based positioning and inertial sensors is leading to the development of new algorithms, that combine these data to produce reliable values, allowing a more general use of this information.

A common control theory tool, the Extended Kalman Filter, has been applied multiple times for this specific purpose, but different approaches can lead to improvements in terms of computational cost and precision. The presented estimation methods are built around two different theoretical backgrounds: a common control technique for nonlinear systems called feedback linearization, and a recently developed parameter identification technique for nonlinear systems, based on the Linear Fractional Transformation (LFT). Both approaches are explained in full and used to build multiple estimators, that are then tested against data sets from a properly-simulated environment and a real unmanned ground vehicle, even in a complete non-adherence situation called drifting. The robustness of each case is analyzed, with the aim of providing high-bandwidth supporting tools for an assisted control solution. Finally, the difficulties linked to the implementation of the algorithms on an on-board microcontroller are treated.

KEYWORDS: Slip Angle; Sideslip Angle; LFT; Linear Fractional Transform; Feedback Linearization; IMU; GPS; GNSS; Lateral control; UGV; EKF.

Contents

Chapter 1

Introduction

1.1 Motivations

Control theory applications are rising exponentially in both industrial and civilian fields, as the performance requested to control systems in terms of precision and bandwidth. This is leading to an increase in the research activity, aimed at optimizing the underlying theory and finding faster and cheaper way to achieve results.

The automotive industry is of particular importance, since control solutions of increasing complexity are nowadays a fundamental part of a vehicle circuitry, for safety, simplicity or luxury purposes. A commercial car is shipped with up to 100 microcontroller units (MCU), each one including its annexed sensors and actuators, that are responsible of a wide variety of tasks at multiple levels, from ignition to mandatory safety systems like ABS and ESP, from active suspensions to collision avoidance systems, a market evaluated over 32 billions in 2017 [28]. In particular, so called driver assistance systems (DAS) are a relevant part of the added value of a car, consisting in features that provide processor-aided assistance to a driver in standard or critical conditions: solutions have been developed for driving with low adherence, avoiding collisions and even easing lane change and parking. These procedures represent an intermediate step between full manual control and autonomous driving.

Stability control is one of the most sensitive task of driver assistance systems, as it consists in preventing the vehicle from spinning, drifting and rolling over [34], or even guaranteeing controllability in these situations, an operation whose efficacy depends on the accuracy and responsiveness of the control algorithms, requiring a much lower response time than the one of non-critical features. In particular, yaw stability refers to the ability of keeping the car on the desired trajectory by monitoring the lateral dynamics, i.e. the lateral forces and displacements that influence the trajectory. While the phenomenon can be modeled in many ways, a common characteristic component is the slip angle, defined as the offset between

the wheels direction and the real direction followed by a vehicle, that arises during a maneuver. The slip angle is an indicator of slippage, a complex phenomenon that is due to tires deformation and non-perfect adherence between tires and road. Its value can be used for assessing the adherence, thus avoiding dangerous situations, and even to track the whole lateral dynamics, allowing to control a vehicle even in non-adherence conditions. Slip angle is only available in critical situations, since its direct measurement requires expensive dedicated sensors, using an optical or inertial working principle.

The spread of low cost positioning and inertial sensors, shipped in form of MEMS devices, like the one represented in Figure 1.1, together with microcontrollers of increasing power, is leading to the development of new safety systems that keep into account both the new inputs and the bigger computational capability, creating a whole new level of automation for standard cars and unmanned ground vehicles. In particular, by introducing sufficiently powerful microcontrollers into a vehicle, it is possible to simulate in real time dynamic models, differential equations that take into account measurements from multiple sensors and provide a much more complete representation of the phenomenon of interest than the one provided by single sensors.

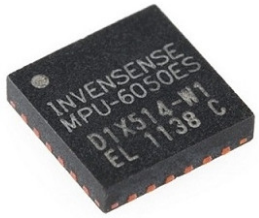


Figure 1.1: The MPU6050, a common 6-dof IMU

The slip angle is not immune to the process, as it can be measured indirectly by feeding data from multiple sensors (generally positioning and inertial sensors) into a dynamic model, making the value available as input for standard safety features or even control in extreme conditions. This task is unfortunately subjected to a series of limitations, shared with many other control theory applications to a real world scenario:

- The accuracy of the chosen model influences significantly the precision of the system. It can be enhanced by increasing the model complexity, requiring more sensors and more computational power.
- Dynamic models of sufficient complexity are generally nonlinear and contains uncertainties, e.g. unknown or time-varying parameters, and are not covered by the traditional approach to the matter. Many methods have been developed to simulate nonlinear models and estimate their uncertainties, the most

common being the Extended Kalman Filter (EKF), a consolidated control theory tool. The EKF has been applied multiple times to the lateral vehicle dynamics field [37] [24] [40] [32], but is subjected to serious limitations in terms of response time, accuracy and computational power requirements.

- The available sensors have a limited precision and bandwidth. These characteristics can worse or improve when data from multiple sensors is processed at the same time (e.g. when using a sensor fusion technique [36]), and must be kept into account. Generally, any solution must be tailored to the specific problem.

These issues are are addressed by a constant research activity: multiple approaches have been proposed for the slip angle estimation problem, and in general for the task of using nonlinear or uncertain models for high-bandwidth applications.

In this thesis, a series of slip angle estimation methods are analyzed and tested against data provided by common sensors, with the aim of implementing practical algorithms that can be used as supporting tools for real-time control systems.

The existing approaches are briefly explained and categorized; in particular, two existing methods are resumed, with the aim of comparing their results to the ones produced by newer approaches:

- An estimation technique based on the application of the EKF on a vehicle dynamics model called single track, fed by measurements from a positioning sensor. This approach constitutes one of the most common setup used in the sideslip estimation field.

- An estimation technique based on the Linear Fractional Transformation (LFT) formalism, a recently-developed technique for parameter identification, that makes use of measurements from a positioning sensor, applied to the single track model to identify up to two uncertain parameters. [8] [10]

The main limitations of these techniques are treated and used as starting point to build and implement two new approaches:

- An estimation technique developed around the Feedback Linerization technique, based on measurements from a positioning sensor. This approach, partially developed in literature [27], is improved and tested against data from a simulated scenario and a real world use case.

- An estimation technique based on the LFT formalism, applied to the single track model, fed by measurements from both a positioning and a IMU sensor, and characterized by up to four uncertain parameters.

Advantages and disadvantages in terms of resolution and robustness are then discussed and supported with available results. The new tools constitute an improvement in terms of precision, noise sensitivity and computational power requirements with respect to some of the approaches proposed by the existing literature. In particular, the tools are intended to partially compensate the problems linked to

the use of GNSS-based positioning sensors, almost mandatory in open spaces but characterized by a low bandwidth and low resolution. At the same time, the work demonstrates that the LFT-based parameter identification procedure for nonlinear systems constitutes an efficient alternative to existing tools.

1.2 Organization

The work is organized in the following chapters:

Chapter 2 is dedicated to the state of the art regarding lateral control, dynamics tracking and slip angle measurement or estimation. Furthermore, a brief explanation of the used sensors is provided.

Chapter 3 explains the lateral behavior of a vehicle during a maneuver and the mathematical model derived to reproduce it; a common tyre model, called Pacejka model, is explained, and a simplified model called single track is derived through successive approximations.

Chapter 4 explains the canonical application of the Extended Kalman Filter to the lateral vehicle dynamics field: a tool is developed to identify the uncertain parameters of the single track model, resulting in a working estimator. The algorithm is then tested against available data.

Chapter 5 is dedicated to the slip angle estimation method derived from the application of the feedback linearization control technique on the unicycle model. Working principle of feedback linearization is explained, as its application to vehicle dynamics. The estimator is then tested against experimental data and upgraded to provide a sufficient degree of robustness.

Chapter 6 is dedicated to a first application of the LFT estimator on the vehicle dynamics. The Linear Fractional Transformation is introduced, together with one of its possible applications to the static minimization problem, then the procedure is applied on the single track model to derive a first estimator, that works through a positioning sensors. The tool is then tested against simulated data and results are analyzed.

Chapter 7 is dedicated to the possible improvements of the LFT-based slip angle estimator. Three possibilities are analyzed and tested: the introduction of data from IMU, an additional variability of the model and the repeated estimation over time. Finally, the LFT-based estimation is analyzed and suggestions for future improvements are given.

Chapter 2

State of the art

2.1 Stability control

As already introduced, an important group of driver-assistance systems is dedicated to driving stability, aimed at increasing safety or ride quality by controlling the vehicle dynamics. Safety solutions are developed around the detection of dangerous situations, while ride quality solutions focuses on the compensation of vibrations or deviations from the desired path. The need for better safety systems is the reason that has led to the introduction of controllers into the automotive field: the Anti-lock braking system (ABS) was introduced in the 70's, consisting in wheel speed sensors, hydraulic valves and a controller able to detect excessive differences between a wheel velocity and the vehicle velocity, thus modulating the braking to avoid a loss of longitudinal adherence. Regarding lateral dynamics, the most advertised assisted control solution is probably the Electronic Stability Control (ESC), introduced in the 90's and mandatory in the European Union from 2014, that monitors the deviation between the vehicle velocity direction and the direction imposed by the steering wheel, braking automatically in case of excessive loss of control. Nowadays, stabilizing systems are available for controlling almost every part of vehicle dynamics. A possible categorization can be performed by dividing the systems with respect to the controlled axis of a vehicle reference frame [34]:

- Longitudinal safety and control systems, that acts on the longitudinal vehicle axis, like ABS or cruise control.
- Lateral safety and control systems, that focuses on lateral forces or steering, like ESC and more recently yaw control systems and lane departure avoidance [29].
- Vertical safety and control systems, like active suspensions and anti-vibration solutions.

Of course modern cars are shipped not only with stabilizing solutions, but also with

advanced driver assistance solutions (ADAS) that automatize a series of complex tasks, like parking, avoiding collisions and protecting pedestrians, systems that work at a higher level and at a slower speed than the arguments considered in this work, and therefore can work properly only if lower level controls have a faster response.

This world of mechatronic devices requires a precise tracking of the vehicle dynamics, a task traditionally performed on race cars and now required on standard cars, that like all low-end products cannot be shipped with complex or expensive components.

2.2 The slip angle

We focus on lateral control solutions, that monitors the lateral forces or displacements for safety or control purposes. A part of these systems rely on the ability of knowing fully or partially the kinematic and dynamic quantities of a vehicle in real-time, e.g. position, velocity and applied forces of its center of gravity or tires: the more data is available, the more the possibilities are.

Since a ground vehicle can be modeled as a relatively simple mechanism, with a low number of inputs (a standard car has just 3 pedals and a wheel), the problem of tracking its kinematic can appear trivial: we could, for instance, measure the velocity and the steering wheel position, and use the dead reckoning method to predict the car position over time. Experimental results show that this approach leads to increasing errors [16], not only due to the drift (accumulation of measurement errors over time), but also due to the non-perfect adherence of the wheels with the terrain, caused by the road conditions and the strain of the tires, a phenomenon difficult to characterize that translates into a lateral force and an offset between the desired direction, imposed by the wheels direction, and the real car direction, generally indicated with the term slip angle.

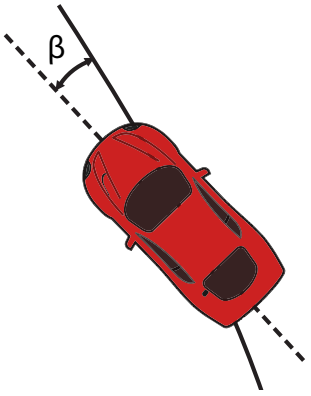


Figure 2.1: Deviation from desired trajectory (dotted) during a maneuver

In literature, the slip angle relative to a single wheel is called wheel slip angle and referred as α, while the sleep angle relative to the entire vehicle is called sideslip angle, and referred as β [31] [19]. The experience shows that this value increases with the velocity and the steering angle, but also decreases while the adherence with the terrain increases. In practice, it is a direct consequence of the stress sustained by the tires, that can be modeled as a lateral force applied at the contact point of the tires or at the center of gravity of the vehicle. In particular, two working zones can be identified: one in which β rises linearly with the lateral force, typical of low slip angle values, and one in which β rises independently of the lateral force, a sign that the adherence limit has been reached. The ratio between the force and the slip angle depends on a series of factors, and its behavior will be explained in the vehicle dynamics chapter.

The measurement or estimation of β allows to correctly reconstruct the vehicle lateral dynamics, by simply summing up it to the wheel direction, obtaining the real vehicle direction. Another value that is strictly linked to the slip angle is the road friction coefficient (μ), that taken alone cannot constitute a link with the dynamic quantities, but can help the reconstruction of β and is often estimated concurrently.

Given these considerations, there are many reasons to equip a vehicle with a slip angle measurement system:

- to evaluate the tires performance, with respect to the road condition and sustained stress.
- to detect dangerous situations, like skidding (loss of traction), and to implement advanced safety solutions that make use of the entire vehicle dynamics.
- to implement lateral control solutions, providing them with the needed dynamical quantities.

2.3 Slip angle measurement

Many methods have been proposed to measure or estimate the slip angle, explained in literature or implemented in commercial products. Sideslip angle and lateral velocity can be measured directly only with dedicated optical sensors, unavailable in most situations, a reason that has led to the development of a series of estimation techniques over the year, partially based on control theory approaches, that combine data from available sensors to perform an indirect measurement. A comprehensive list cannot be compiled with precision, although a basic categorization can be tried, based on the working principle of the systems, even if some state-of-the-art techniques use combinations of multiple approaches.

The choice of one or another measurement system depends on budget and available space (a dedicated sensor cannot generally fit on a small unmanned ground vehicle), and also by the needed bandwidth and accuracy. Direct optical sensors

have the highest resolution, and are used for precise tests or performance monitoring. This level of precision is not needed on safety systems and is redundant for some control algorithms.

Estimation techniques have generally a limited range and a low accuracy, a fact that constitutes the main reason for the continuated research activity around them. This difficulty is outlined in the normative dedicated to the evaluation of the dynamic behavior of a vehicle, that defines as acceptable an absolute error of $0.5°$ in sideslip estimation [23], a shift not negligible when compared to β in standard situations ($0 - 2°$ [19]), leading to a relevant relative error. Therefore, this kind of systems are generally not fit for performance monitoring.

All estimation methods are based on sensors that measure position, inertial quantities (like acceleration and angular rate) or velocity, in one or multiple points of the vehicle chassis. Available data is then elaborated in a dynamical system to perform the estimation. Bandwidth and accuracy greatly depend not only on the chosen approach, but also on the sensors characteristics. This constitutes the main limitation, as these systems are highly sensitive to measurement errors, but also a possibility of improvement: it is possible to reuse already-developed algorithms with newer sensors, obtaining better results without modifying the underlying algorithm.

Optical sensors

The industry standard method to directly measure a slip angle consists in using a contact-less optical correlation sensor, placed on the desired point, on the chassis or on the tyre, and facing the road. Infrared light beams are used to measure the longitudinal and vertical velocities [24], and wheel slip angle (or vehicle sideslip angle) is obtained through its definition

$$\alpha = \arctan\left(\frac{v}{u}\right) \tag{2.1}$$

where v is the lateral velocity, and u the longitudinal velocity. Typical resolution is below $0.01°$, with a bandwidth up to $250\ Hz$ [1]. As already mentioned, the problem of this system is its cost (thousands of dollars) and the need of mounting a visible, bulky and sensitive sensor, a task often performed during tests of race cars, but seldom in other contexts.

Position-based estimators

A first family of estimators (GPS/INS) process data from a single satellite-based positioning sensor and an Inertial Navigation System (INS), placed at the point of interest, to obtain measurements of position, lateral acceleration and yaw rate. Position values are updated with INS values through an arbitrary method, the most common being the averaging observer or the Extended Kalman Filter (EKF)

[37], an important tool that will be treated in Chapter 4. The slip angle is then found by simply computing the difference between the vehicle attitude γ and the velocity direction (yaw angle) ψ

$$\beta = \gamma - \psi \tag{2.2}$$

By using a high-resolution GPS implementation (like Real Time Kinematic) it is possible to obtain slip angle resolutions of $0.1°$, and a bandwidth of almost $10\ Hz$ [24]. The sensors and processing unit are usually shipped into a single, portable box, that is placed inside the vehicle.

Another estimation technique that can fall into this category is the one that uses two GPS sensors, placed at opposite sides of the vehicle: this method results in a much more precise computation of the vehicle direction [4], that is otherwise subjected to drift and bias errors. Slip angle computation is performed in a similar way to the previous case.

Dynamic model-based estimators

The performance of position-based estimators can be enhanced by adopting a model of the vehicle lateral dynamics, a representation of the phenomenon through differential equations, that links together the slip angle with lateral forces, attitude and position. It is then possible to adopt one of the many control theory approaches that allow observation of non-measured state quantities by relying on a previous knowledge of the system model. Multiple techniques for linear or nonlinear models can be used to minimize the error between data from available sensors and the output of a virtual model, that has the slip angle in its state variables.

Observers and Extended Kalman Filters are the standard tools to perform this task. An alternative technique consists in using the feedback linearization procedure in a closed loop, obtaining an observer without the approximation typical of linear systems, a procedure initially developed for stabilizing robot movements [27], that will be analyzed in this work.

Parameter identification-based estimators

Model-based estimators can be improved by introducing uncertain parameters in the equations. This allows a further modeling of the car lateral dynamics, highly dependent on road and tire conditions. Model parameters can be identified progressively or in a single step, and the resulting system can be used for predictive estimations.

Parameter identification is a recurring problem in engineering, and multiple methods have been developed for this purpose. The most used is the Kalman Filter and its evolutions for nonlinear models (Extended Kalman Filter (EKF), unscented EKF, hybrid EKF).

This approach has two main limitations. The first is that the considered models have inherent uncertainties [19], and can be used only with specific assumptions and for a limited timespan, after which parameters must be updated. The second is that these equations are highly nonlinear, so either a linearization or a method for nonlinear systems must be used. Computational power requirements must also be taken into account when dealing with nonlinear systems.

The Kalman Filter for linear systems and the linearization procedure are still used in some sideslip estimators, when there are computational cost constraints or in special cases, like articulated heavy vehicles, that require a complex model [11] [30].

A relevant part of these approaches are based on the application of the EKF on a variant of the single track model, a simplified vehicle dynamics model unanimously used [32]: the drawback is its excessive computational cost with respect to alternative methods. Among the various works, an interesting proposal consists in using a two-step procedure, that parses inertial measurements and velocities to estimate the tire lateral forces, and then lateral forces to estimate the slip angle through the single track model [20], analyzed in a previous thesis work [10].

An evolution in terms of precision can be provided by the addition of a tyre model, like the Pacekja magic formula, a semi-empirical formula that links the lateral force to the slip angle and other quantities through a series of parameters [31], that must be properly identified. An improved tool, using the EKF and Pacejka model, can be built by placing accelerometers on the tires, measuring directly their force [40], an option not suitable for common vehicles. Among the proposals that do not make use of the EKF, an interesting approach constists in using a two-stage (TS) method for parameter identification, alternative to the EKF and, according to its authors, more efficient [17].

Unscented EKF has been applied with success to alternative tyre models (Dugoff tyre model) for sideslip estimation [14], another example of the possibilities offered by a model-based approach.

Another technique alternative to the EKF is based on a formalism called Linear Fractional Transformation (LFT), that separates respectively the uncertainties, the linear and nonlinear parts of a nonlinear model, and its exploitation for parameter identification. The method has been applied multiple times to identify the single track model parameters by using position measurements [8], and can be improved with a proper validation.

Training-based estimators

The rise of self-learning computing algorithms, the most famous being the neural network, is leading to their application in multiple fields, even in the parameter identification procedure. A system can progressively learn how to identify and estimate the sideslip angle and the other vehicle dynamic quantities, starting from

ground or with the help of a model. This approach has been proposed in literature and partially tested [10] [18], resulting in tools which produce discrete results, although with higher response times and computational power requirements than their model-based counterparts.

2.4 Position and inertial sensors

Before introducing the theoretical part, a brief explanation of the sensors referred in the work is provided. The efficacy of sideslip estimation techniques greatly depends on sensor characteristics, and they have a predominant role in the final accuracy of the results: therefore, knowing their limitations in terms of resolution and bandwidth allows an evaluation on what we can expect from a certain final tool, giving its sensitivity to measurements. Furthermore, data from multiple sensors can be combined to strengthen a certain measurement, a technique known as sensor fusion.

Position sensors

As a position sensor we refer to a device that measures the position of an object through numerical values. In case of vehicles, we are interested in contact-less devices that allow the measurement of the chassis planar displacement with respect to an external reference frame, or with respect to the geographic coordinates (longitude and latitude). The main benefit of using data from a sensor that refers to an external point is that it allows to compensate progressive drift errors typical of devices that use the vehicle itself as a reference frame, like accelerometers and gyroscopes.

A series of sensors have been developed for position measurement in closed or limited spaces, based on lasers, ultrasounds and cameras, generally not suitable for a common usage. The data used to validate the proposed estimators comes from the tracking of a vehicle through an optitrack system, a commercial name for a camera-based tracking system, that processes data from multiple cameras to detect a marker position. Data can then be transmitted to a processor or to an on-board controller in various way, for instance, through the use of a Wireless Local Area Network (WLAN).

The only way to continuously track an item position in a general, open space, consists in using an on-board Global Navigation Satellite System (GNSS) sensor. GNSS technology, in development since the 60's, nowadays is based on the presence of a set of 10-30 satellites in multiple orbits, that periodically emits digital radio signals, containing their position and the time of the message. Sensors are able to receive signals from visible satellites and to compute the time difference between the signals, obtaining the geographic coordinates.

Available GNSSs are developed and maintained by the main world superpowers for military and civil applications: Global Positioning System (GPS) is owned by the United States, GLONASS by Russia, Galileo by the European Union and BeiDou by China. GPS is the the most famous, often used as an acronym for GNSS, but since 2017 most of the alternative systems offer a global coverage, although they require capable sensors.

Civilian GNSS receivers are usually characterized by low resolutions ($1 - 3\ m$) and bandwidth ($1 - 10\ Hz$), and cannot be used everywhere due to their working principle, as the signal cannot be received clearly in closed places.

In particular, signals are subjected to particular errors: while the noise is usually limited [36], the presence of obstacles capable of reflecting radio signals, like woods, mountains or big constructions, causes the wrong computation of the signal timing, that translates in sudden shifts in the measurements.

A series of improvements has been developed over the years and can significantly influence the system performances, leading to their introduction in some of the GNSS-based commercial sideslip sensors:

- Real Time Kinematic (RTK) is a GPS enhancement technique that relies on the additional measurement of the signal phase, and uses an additional ground receiving station to compute the phase difference, increasing resolution up to $0.01\ m$.
- Various augmentation techniques have been developed, using additional ground-based and satellite-based signals, available on local basis (WAAS, EGNOS, SDCM), to improve the position measurement.
- The capability of scanning available Wi-Fi hotspots, together with a database of locations of known wireless access points, is often used to improve the GPS measurement or to totally replace it (Wi-Fi positioning systems).

GNSS sensors are set to become more and more precise in the future, and their low cost and diffusion make them considerable for current and future control theory applications.

IMU sensors

An Inertial Measurement Unit (IMU) is an electronic system that encapsulates multiple inertial sensors, in particular accelerometers, gyroscopes and magnetometers. An accelerometer is a device that measures the acceleration along a specific axis; a gyroscope is a device that measures the angular rate around a specific axis; a magnetometer is a device that measures the magnetic field along a specific axis.

IMU platforms were developed in the 50's for space and aerial applications, but are now produced in the form of Micro Electro Mechanical Systems (MEMS) and shipped with low-end products, like smartphones and drones.

An IMU can contain a variable number of accelerometers, gyroscopes and magnetometers, placed on different axes; in this work a 6 degrees-of-freedom IMU is considered, consisting of 3 accelerometers on 3 independent axes, and 3 gyroscopes on 3 independent axes. Regarding sideslip estimation, only the yaw rate (angular rate around vertical axis) and the lateral acceleration are significant.

The main problem with modern MEMS IMU is that their measurements are characterized not only by a strong gaussian noise, but also by a significant bias error, generally not removable by acting on the sensor position. An electronic calibration is required, manually or through an automatic algorithm [33], along with a periodic check, since any change to the setup can lead to new errors.

The high noise level constitute a problem also when integrating values from these kind of sensors, e.g. integrating the acceleration to obtain the speed or the yaw rate to obtain the attitude, a practice sometimes performed but generally discouraged, as errors do accumulate over time. Data generated from integrations is generally corrected through a merging with data from sensors that provide an absolute reference, like a positioning sensor.

The orientation, which is a useful quantity in the vehicle dynamics field, can also be computed from a magnetometer, but its use has been discarded, due to the sensitivity of this kind of sensor to external magnetic fields, that can critically influence the robustness of the algorithm. In previous works it was demonstrated that magnetometer measurements are not reliable and have to be processed through a specific filter before being used in a sideslip estimator [42].

Chapter 3

Lateral Vehicle Dynamics

In this chapter the basic behavior of the vehicle dynamics, restricted to the lateral axis, is introduced and progressively modeled, up to two common systems of differential equations used by the estimators presented in this work, the unicycle and the single track model. Particular emphasis is given to the approximations performed during each step, that allow to predict the working range in which the estimators will work and offer a future possibility of improvement, as they are associated to the uncertainties that are the main limitation of model-based estimators.

3.1 Pacejka tyre model

Dynamic modeling for any ground vehicle starts from tires, whose characteristics are of crucial importance in determining the overall vehicle behavior. The interface between a rolling tyre and the road is usually denoted as single contact point, but in reality we must think of it as a contact patch, with multiple points in contact between the two surfaces. Inside this patch, tire forces are generated due to two reasons: the friction in the patch and the elastic deformation of the tyre [15].

The friction phenomenon is seldom analyzed directly due to complexity reasons, but is instead modeled with coefficients and formulas. Tires makes no exception, and a series of model has been developed, based on simple physical considerations or experimental data. We are interested in dynamic models, able to represent both the transient and the steady-state part of the response. The most used and famous tyre model is the one developed by Pacejka, explained in dedicated books [31], available in simulation environments (here Dymola) and even video games (e.g. in the popular Unity Engine). An introduction to Pacekja model is provided, as its quantities are used throughout the work and are at the base of the common single track model.

A rolling tyre in contact with the road can be modeled as in Figure 3.1, where the following quantities are defined:

- α is the wheel slip angle.
- F_x, F_y and M_z are the forces and aligning torque that arise at the contact patch with the road, respectively F_z is the reaction force due to the vertical load (mass of the vehicle).
- V is the velocity vector of the wheel, pointing in the direction imposed by the slip angle; v and u are the projections of V on the lateral and longitudinal axes.
- Ω_0 is the rotation speed of the wheel.

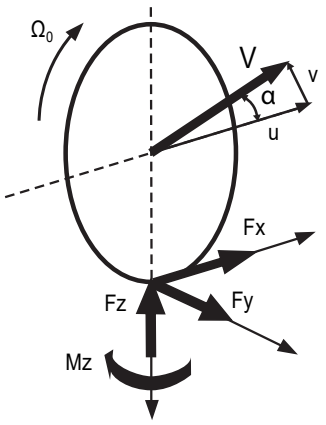


Figure 3.1: Rolling tyre and quantities involved

Lateral slip angle can be defined as the ratio between the two velocity components

$$\alpha = \arctan\left(-\frac{v}{u}\right) \tag{3.1}$$

Other quantities used in literature are the longitudinal slip κ, that arises in cases of an applied torque on the wheel, and the camber angle γ, which is the inclination of the wheel with respect to the road. Forces and aligning torque can be modeled as functions of just α, κ, γ and F_z

$$F_x = F_x(\kappa, \alpha, \gamma, F_z) \tag{3.2}$$
$$F_y = F_y(\kappa, \alpha, \gamma, F_z) \tag{3.3}$$
$$M_z = M_z(\kappa, \alpha, \gamma, F_z) \tag{3.4}$$

The influence of the arguments on each quantity has been determined experimentally; in particular, the characteristic shape of α versus F_y and M_z, considering $\kappa = 0, \gamma = 0$ is shown in Figure 3.2. Of course a nonzero longitudinal slip (typical of braking) or camber angle influence the shape of the curve, but their effect is considered separately.

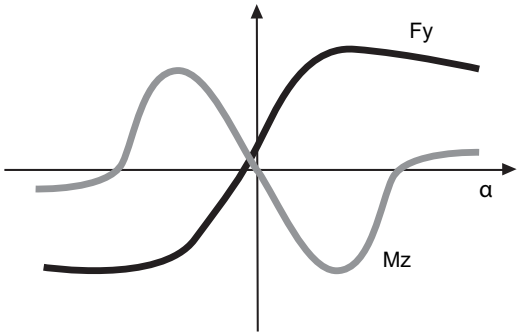


Figure 3.2: Slip angle influence on lateral force and aligning torque

The slope of each of these "pure slip" curves is called stiffness, and denoted with the letter C. Using this definition, a linearization of the force and moment expressions leads to this result

$$F_x = C_{f\kappa}\kappa \tag{3.5}$$
$$F_y = C_{F\alpha}\alpha + C_{f\gamma}\gamma \tag{3.6}$$
$$M_z = -C_{M\alpha}\alpha + C_{M\gamma}\gamma \tag{3.7}$$

Focusing the on lateral slip stiffness (cornering stiffness $C_{f\alpha}$), experimental results have shown that

- $C_{f\alpha}$ is not influenced by road conditions and longitudinal speed at low slip angles, while at high angles both factors concur to decrease the slope.
- $C_{f\alpha}$ is highly influenced by the load F_z and the vehicle type (articulated heavy vehicles require a different approach).

This leads to conclude that the cornering stiffness value depends on the specific vehicle and the specific conditions, but is almost constant for low slip angle values, an assumption that is at the base of some estimators.

The showed curve can be modeled with the semi-empirical *Pacekja Magic formula*, whose parameters, explained in dedicated literature, depends on the tyre size, constitutive material, inflation, pressure, deterioration and other properties, and are usually estimated from experimental data [17]. The F_y and C_f formulas are reported below, to show the α and F_z influence.

$$F_y = D\sin[C\arctan[B\alpha - E(B\alpha - \arctan(B\alpha)]] \tag{3.8}$$
$$F_{y,peak} = D = \mu F_z \tag{3.9}$$
$$C_{f\alpha} = BCD = c_1\sin[2\arctan(\frac{F_z}{c_2})] \tag{3.10}$$

3.2 Single track model

Considerations on tire behavior could be used to directly estimate the slip angles of each wheel, given the possibility of placing a series of sensors on each single tyre, an operation impossible for small ground vehicles and not suited for commercial cars. An option consists in linking the tires dynamic quantities with the one of the vehicle, represented as a concentrated body in its center of gravity (COG). The tyre model can be transformed in a full ground vehicle model with the following assumptions:

- track width is negligible with respect to the circular motion radius;
- steer and slip angle assume low values, so it is possible to use trigonometric approximations like $\cos\alpha \approx 1$, $\sin\alpha \approx \alpha$, $\tan\alpha \approx \alpha$;
- roll moment remains small;
- speed of travel is considered to be almost constant; theory can hold in case of quasy-steady-state situations like moderate braking [31];
- lateral component of longitudinal forces F_x are neglected;
- the contact points of the tires is influenced by braking or accelerating, and it shifts rearwards or backwards; this is neglected;

The first hypothesis allows to neglect the car width, making possible the modeling of a standard 4-wheel vehicle with a two-wheel vehicle, as in Figure 3.3, upon which it is possible to build the single track model (or bycicle model), originally proposed in the 40's [29] and still used today in most of the sideslip estimation techniques.

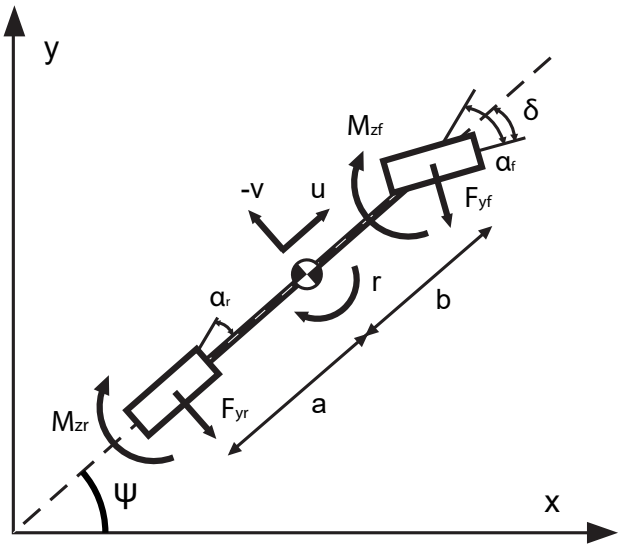


Figure 3.3: Single track model

The following terms are defined:

- m and I_z are the mass and inertia of the chassis;
- u and v are the longitudinal and lateral component of the chassis velocity vector;
- a and b are the distances of the rear and front tyre axes from the center of mass;
- F_y and M_z are the lateral force and aligning torque present at the contact patch of the two virtual tires;
- α_r, α_f are the slip angles of the tires;
- δ is the steering angle, applied only to the front tyre;
- r is the yaw rate;
- x, y are the cartesian coordinates of the vehicle center of mass, and ψ its orientation with respect to the origin;

Two reference frames are used, one moving with the vehicle center of gravity and one fixed. The moving frame is used to define the velocity, through its components v (lateral) and u (longitudinal), and yaw rate r, while the fixed frame is used for the cartesian coordinates x, y and the orientation ψ. Relationships between quantities in the two frames are expressed through the standard rotation matrix

$$\dot{x} = u\cos\psi - v\sin\psi \tag{3.11}$$
$$\dot{y} = u\sin\psi + v\cos\psi \tag{3.12}$$
$$\dot{\psi} = r \tag{3.13}$$

These formulas can be used to link the position with the velocity and yaw rate, forming the unicycle model, which is a useful kinematic representation of a moving object and will be used later.

By applying the virtual work principle on Figure 3.3, neglecting higher-order terms and approximating the trigonometric functions, it is possible to obtain these equations of motion

$$m(\dot{v} + ur) = F_{y,f} + F_{y,r} \tag{3.14}$$
$$I_z\dot{r} = aF_{y,f} - bF_{y,r} \tag{3.15}$$

We now need to link the lateral forces to the slip angles, using the provided theory; the simplest way consists in using the previously-defined cornering stiffnesses

$$F_{y,f} = F_{y,f}(\alpha_f) \approx C_f\alpha_f \tag{3.16}$$
$$F_{y,r} = F_{y,r}(\alpha_r) \approx C_r\alpha_r \tag{3.17}$$

Finally, the slip angles, that by definition are the offset between the tires physical directions and their velocities, are linked to the inertial properties of the vehicle COG, and to the steering angle (G is used as steering gain)

$$\alpha_f = G\delta - \frac{1}{u}(v + ar) \tag{3.18}$$

$$\alpha_r = -\frac{1}{u}(v - br) \tag{3.19}$$

The overall vehicle sideslip angle β can then be computed through its definition.

The complete single track model is given by

$$\begin{cases} m(\dot{v} + ur) = F_{y,f} + F_{y,r} \\ I_z\dot{r} = aF_{y,f} - bF_{y,r} \\ F_{y,f} = C_f\alpha_f \\ F_{y,r} = C_r\alpha_r \\ \alpha_f = G\delta - \frac{1}{u}(v + ar) \\ \alpha_r = -\frac{1}{u}(v - br) \\ \dot{\psi} = r \\ \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \\ \beta = \arctan(\frac{v}{u}) \end{cases} \tag{3.20}$$

An alternative formulation, used in previous works, can be obtained by replacing

$$u = v\cos\beta \tag{3.21}$$

resulting in this system

$$\begin{cases} mv\cos(\beta)\ (\dot{\beta} + r) = F_{y,f} + F_{y,r} \\ I_z\dot{r} = aF_{y,f} - bF_{y,r} \\ F_{y,f} = C_f\alpha_f \\ F_{y,r} = C_r\alpha_r \\ \alpha_f = -\beta - \dfrac{ar}{v\cos\beta} + G\delta \\ \alpha_r = -\beta + \dfrac{br}{v\cos\beta} \\ \dot{\psi} = r \\ \dot{x} = v\cos(\psi + \beta) \\ \dot{y} = v\sin(\psi + \beta) \end{cases} \tag{3.22}$$

Steering angle δ and longitudinal velocity u are usually considered as inputs; u, as already seen, has an influence on C_f, so it must not be subjected to excessive variations. Also δ influences C_f, since it is part of the front tyre slip angle.

The model validity range depends on how the various parameters are handled:

- Cornering stiffnesses can be considered as constant parameters, allowing the use of simple identification techniques, but this assumption narrows the model operating range in the low slip angle zone ($\alpha < 5 \div 10^\circ$). Alternatively, stiffnesses can be modeled through a semi-empirical formula, but in this way more uncertain parameters have to be identified.
- The vehicle mass and inertia could be subjected to small variations, due to vehicle loading or longitudinal stress (neglected here): their use as constants could be a limiting factor.

It is also important to observe the presence of the longitudinal velocity in some denominators, that creates a singularity for $u \to 0$. This limitation is typical of vehicle dynamic models and must be handled properly in implementations, for instance by detecting the problem and resetting the system before incurring in a critical error.

Chapter 4

Slip angle estimation through the Extended Kalman Filter

This chapter is dedicated to an implementation of a sideslip estimator based on the Extended Kalman Filter, the tool upon which a great quantity of estimation techniques have been developed in both the academic and industrial world, as mentioned in the state of the art chapter. The algorithm is first briefly explained, then an estimator is developed around the single track model and implemented with the available software. Tests are performed by using data sets from a simulated environment and a real UGV, with the aim of producing results that can be compared to the ones of the new estimators presented in this work.

4.1 The Extended Kalman Filter

The Kalman Filter and its variants (Extended Kalman Filter (EKF), Unscented EKF, Hybrid EKF), are tools that allow to exploit available measurements to increase the precision of measured quantities or estimate non-measured quantities, through the use of a dynamic model. Despite their age, these algorithms remain along the most used tools in the control and estimation field, due to their simplicity and efficacy. In particular, they represent the standard approach in the sideslip estimation field, since they can adapt very well to common and uncommon situations without the need of changing the underlying math. A great quantity of academic material have been produced in the past years [19] [14] [5] [41] [24] [4], with results that are still competitive with the ones of newer techniques in terms of robustness and precision.

A detailed explanations of the Kalman Filter is available in dedicated literature [39]; in this work, we are not interested in covering all the aspects and variants of the tool, but in explaining briefly the theory related to a specific case (when the considered system is nonlinear and the EKF-variant is used) and using this knowledge to implement an already consolidated sideslip estimator.

We first consider a linear, time-varying, continuous system in the form

$$\begin{cases} \dot{x}(t) = F(t)x(t) + G(t)u(t) + L(t)w(t) \\ y(t) = H(t)x(t) \end{cases} \tag{4.1}$$

where $x(t)$ is the model state vector, $y(t)$ the output vector, $u(t)$ the input vector, $w(t)$ a gaussian noise and $F(t)$, $G(t)$, $L(t)$, $H(t)$ are time-varying matrices.

We can observe the system output y only throughout a set of measurements z, that also include a gaussian noise n

$$z(t) = y(t) + n(t) = H(t)x(t) + n(t) \tag{4.2}$$

The initial state and noise vectors at each step are assumed to be mutually independent.

We suppose the available measurements z are "corrupted", i.e. they are very noisy, or there is a state variable in vector x that is not measured directly.

We want to compute an estimate of the full state $\hat{x}(t)$ and the output $\hat{y}(t) = H(t)\hat{x}$, such that the result is coherent with the available measurements (the hat denotes an estimation). We can define a measurement residual error

$$\varepsilon(t) = z(t) - H(t)\hat{x}(t) \tag{4.3}$$

and minimize it (iteratively finding its minimum) by minimizing a weighted quadratic cost function

$$J(t) = \frac{1}{2}\varepsilon(t)^T S^{-1}\varepsilon(t) = \frac{1}{2}[z(t) - H(t)\hat{x}(t)]^T S^{-1}[z(t) - H(t)\hat{x}(t)] \tag{4.4}$$

where S is a weighting matrix, usually chosen to weight the different covariances of z and x.

The problem of this method is that the minimization can be performed only after all the measurements have been collected along the entire timespan; it is better to develop a method that can minimize the cost function while measurements are still gathered, and can integrate any future event (input or measurement noise) without doing a complete recalculation.

By considering n as a gaussian noise and propagating the statistic properties (covariance and mean) of both n and x, exploiting the necessary optimality condition and doing a series of intermediate steps, it is possible to transform the problem of minimizing equation (4.4) into the simulation of the following system, that also

consider the evolution of state vector x dictated by the dynamic model (4.1)

$$\begin{cases} \dot{\hat{x}}(t) = F(t)\hat{x}(t) + G(t)u(t) + K_c(t)[z(t) - H(t)\hat{x}(t)] \\ \dot{P}(t) = F(t)P(t) + P(t)F(t)^T + L(t)Q_c'(t)L^T - P(t)H(t)^T R_c^{-1} H(t)P(t) \\ K_c(t) = P(t)H(t)^T R_c^{-1} \\ given\ \hat{x}(0) = x_0 \\ given\ P(0) = P_0 \end{cases} \tag{4.5}$$

where:

- $\hat{x}(t)$ is an estimate of the model state vector, $u(t)$ the input vector (fully known);
- $F(t)$, $G(t)$, $L(t)$, $H(t)$ are time-varying matrices;
- $P(t)$ is the error covariance matrix, that represent the estimation error at the given time t;
- K_c is the gain matrix, calculated through P, that weights the error between the measurements z and the system output.
- R_c is the covariance matrix of the measurement noise;
- Q_c' is the spectral density matrix of the process.

The equation related to matrix P is a Riccati Differential Equation, a standard equation that is a solution to the problem of minimizing a quadratic function, like the one expressed in (4.4). The system above is called *Kalman-Bucy Filter* and is an optimal estimator for linear systems.

Unfortunately the dynamic models used in the vehicle dynamic field are nonlinear, and are in the form

$$\begin{cases} \dot{x}(t) = f[x(t), u(t), w(t), t] \\ y(t) = h[x(t), t] \end{cases} \tag{4.6}$$

where f and h are function vectors; $u(t)$ and $w(t)$ are not assumed to appear in the system output y.

Therefore the base version of the Kalman Bucy filter cannot be used for the scope. A linearization can be performed, at cost of introducing a significant approximation in the results, alternatively the algorithm needs to be upgraded to account for this scenario.

Among the variants of the Kalman Filter designed for nonlinear systems, the Extended Kalman Filter is the most immediate, as it consists in replacing the linear model simulation in first equation of system (4.5) with a nonlinear one, while keeping the error propagation mechanism as it is, replacing the elements of the linear model with approximations, since it can be demonstrated that the

covariance and mean of x and n propagate in a similar way to the linear case [39]. The filter results as follows

$$\begin{cases} \dot{\hat{x}}(t) = f[\hat{x}(t), u(t), w(t), t] + K_c(t)[z(t) - H(t)\hat{x}(t)] \\ \dot{P}(t) = F(t)P(t) + P(t)F(t)^T + L(t)Q_c'(t)L^T - P(t)H(t)^T R_c^{-1} H(t)P(t) \\ K_c(t) = P(t)H(t)^T R_c^{-1} \\ given\ \hat{x}(0) = x_0 \\ given\ P(0) = P_0 \end{cases} \tag{4.7}$$

where $F(t)$, $G(t)$, $L(t)$, $H(t)$ are linearizations of the nonlinear model in the neighborhood of the considered time instant

$$F(t) = \left.\frac{\partial f}{\partial x}\right|_{x(t),u(t),w(t)} \tag{4.8}$$

$$L(t) = \left.\frac{\partial f}{\partial w}\right|_{x(t),u(t),w(t)} \tag{4.9}$$

$$H(t) = \left.\frac{\partial h}{\partial x}\right|_{x(t),u(t),w(t)} \tag{4.10}$$

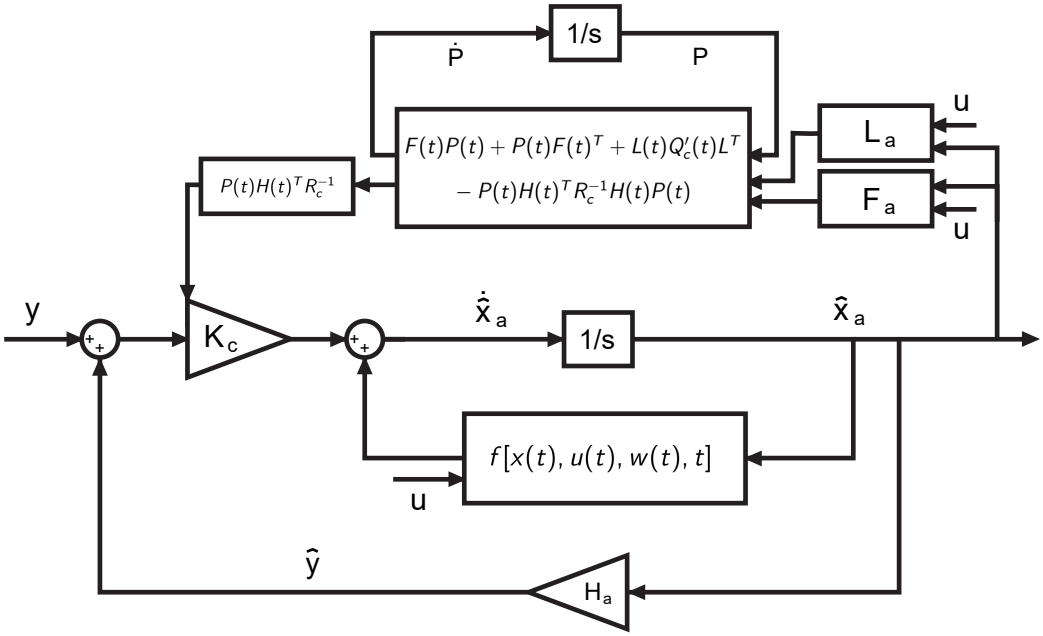


Figure 4.1: Block diagram of the Extended Kalman Filter

The state estimation is not guaranteed to be optimal, and there are convergence and stability problems related to the choice of the initial conditions x_0 and to the value of x over the considered timespan; moreover, the model must be linearized

at each time step, an operation with an high computational cost. Nevertheless, the tool should work if proper values of x_0, P_0, Q'_c and R_c are chosen.

The block diagram of the complete system is reported in Figure 4.1.

4.2 Application

We now have a tool that, given a dynamic model, can estimate a set of state variables from available measurements, an operation that is particularly fit for estimating the sideslip angle, that as already said in the state of the art chapter, is generally not measured directly, but through the merging of data from sensors with a different purpose.

The estimator described in this section represents one of the most typical applications of the EKF to the vehicle dynamic field, and it requires a single positioning sensor and the vehicle input commands to achieve the desired result.

We recall the single track model (equations (3.20)), written in this canonical form

$$\begin{cases} \dot{v} = \frac{1}{m}[C_f(\frac{-v}{u} - a\frac{r}{u} + G\delta) + C_r(\frac{-v}{u} + b\frac{r}{u})] - ur \\ \dot{r} = \frac{1}{I_z}(aC_f(\frac{-v}{u} - a\frac{r}{u} + G\delta) - bC_r(\frac{-v}{u} + b\frac{r}{u})) \\ \dot{\psi} = r \\ \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \end{cases} \tag{4.11}$$

where the state vector x is

$$x = \begin{bmatrix} v & r & \psi & x & y \end{bmatrix}^T \tag{4.12}$$

The longitudinal velocity u is usually measured through an encoder on a vehicle rear wheel, while the steering angle δ is usually read from the control system (in case of an UGV) or measured from the steering wheel position; we assume both values are available as inputs, and insert them in the input vector

$$u = \begin{bmatrix} u & \delta \end{bmatrix}^T \tag{4.13}$$

Given a set of measurements y, an application of the Extended Kalman Filter on the model should allow to estimate the lateral velocity v, that is a state variable, and consequently the sideslip angle, through the formula

$$\beta = \arctan\left(\frac{v}{u}\right) \tag{4.14}$$

Of course matrix Q and R must be properly filled with values that indicate the covariance of respectively the state vector and the measurement noises, as well as

$L = \partial f / \partial w$, that links the noise w to the state vector. Q and L are not needed at the moment, while values proven to be reliable for R, relative to the considered tests, are

$$R = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix} \tag{4.15}$$

where non-diagonal terms are empty because there must be no correlation between noises.

A set of constant parameters is needed to simulate the model. The mass m, the inertia I_z, the distances a, b can be measured before using the tool, and their value assumed constant.

Nevertheless the front and rear cornering stiffnesses C_f and C_r, depend on a series of factors, including the terrain and tire conditions, and cannot be assumed static. Their values cannot be estimated offline, but must be treated as uncertain parameters.

There exist a popular technique which allow to exploit the Extended Kalman Filter to identify uncertain parameters: the first step consists in augmenting the state vector with an additional element for each uncertain parameter, in this case one for C_f and one for C_r

$$x_a = \begin{bmatrix} \bar{x} \\ C_f \\ C_r \end{bmatrix} = \begin{bmatrix} \bar{x} \\ x_6 \\ x_7 \end{bmatrix} \tag{4.16}$$

the original parameters are replaced in the dynamic model with the corresponding augmented state variable; in this case, we obtain

$$\begin{cases} \dot{v} = \frac{1}{m}[x_6(\frac{-v}{u} - a\frac{r}{u} + G\delta) + x_7(\frac{-v}{u} + b\frac{r}{u})] - ur \\ \dot{r} = \frac{1}{I_z}(ax_6(\frac{-v}{u} - a\frac{r}{u} + G\delta) - bx_7(\frac{-v}{u} + b\frac{r}{u})) \\ \dot{\psi} = r \\ \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \end{cases} \tag{4.17}$$

then some fictitious noises v_1 and v_2 are summed to the unknown parameters in the model, obtaining

$$\begin{cases} \dot{v} = \frac{1}{m}[(x_6 + v_1)(\frac{-v}{u} - a\frac{r}{u} + G\delta) + (x_7 + v_2)(\frac{-v}{u} + b\frac{r}{u})] - ur \\ \dot{r} = \frac{1}{I_z}(a(x_6 + v_1)(\frac{-v}{u} - a\frac{r}{u} + G\delta) - b(x_7 + v_2)(\frac{-v}{u} + b\frac{r}{u})) \\ \dot{\psi} = r \\ \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \end{cases} \tag{4.18}$$

these noises are not required in the majority of cases, but they are added as precaution, as the EKF is known to result in wrong estimations when there is not enough noise on the measurements [39].

The model is then augmented with one equation for each uncertain parameter, that imposes the convergence of the parameter value, by setting $\dot{d} = v$ (where d is the parameter and v is another fictitious noise). In this case

$$\begin{cases} \dot{C}_f(t) = v_3(t) \\ \dot{C}_r(t) = v_4(t) \end{cases} \tag{4.19}$$

Finally, matrices F_a, L_a and H_a can be calculated through their definition in equations (4.8)-(4.10), obtaining the complete estimator reported below.

$$x_a = \begin{bmatrix} v & r & \psi & x & y & C_f & C_r \end{bmatrix}^T \tag{4.20}$$

$$u = \begin{bmatrix} u & \delta \end{bmatrix}^T \tag{4.21}$$

$$y = \begin{bmatrix} x & y \end{bmatrix}^T \tag{4.22}$$

$$F_a = \begin{bmatrix} \frac{-x_6}{mu_1} - \frac{x_7}{mu_1} & \frac{-x_6 a}{mu_1} + \frac{x_7 b}{mu_1} - u_1 & 0 & 0 & 0 & \frac{-x_1}{u_1 m} - a\frac{x_2}{u_1} + G\frac{u_2}{m} & \frac{-x_1}{u_1 m} + b\frac{x_2}{u_1} \\ \frac{-ax_6}{I_z u_1} + \frac{bx_7}{I_z u_1} & \frac{-a^2 x_6}{I_z u_1} - \frac{b^2 x_7}{I_z u_1} & 0 & 0 & 0 & \frac{Gau_2}{I_z} - \frac{a^2 x_2}{I_z u_1} - \frac{ax_1}{I_z u_1} & \frac{bx_1}{Izu_1} - \frac{b^2 x_2}{Izu_1} \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -\sin x_3 & 0 & -u_1 sinx_3 - x_1 cosx_3 & 0 & 0 & 0 & 0 \\ \cos x_3 & 0 & u_1 cosx_3 - x_1 sinx_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.23}$$

$$H_a = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.24}$$

$$L_a = [7 \times 4] = \begin{bmatrix} \frac{Gu_2}{m} - \frac{x_1}{mu_1} - \frac{ax_2}{mu_1} & \frac{bx_2}{mu_1} - \frac{x_1}{mu_1} & 0 & 0 \\ \frac{Gau_2}{Iz} - \frac{a^2 x_2}{Izu_1} - \frac{ax_1}{Izu_1} & \frac{bx_1}{Izu_1} - \frac{b^2 x_2}{Izu_1} & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.25}$$

$$R = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix} \tag{4.26}$$

The following covariance matrix was chosen, with each column corresponding to one of the previously-introduced fictitious noises. The non-diagonal terms are

empty because there must be no correlation between noises

$$Q_c' = \begin{bmatrix} 10^3 & 0 & 0 & 0 \\ 0 & 10^3 & 0 & 0 \\ 0 & 0 & 10^3 & 0 \\ 0 & 0 & 0 & 10^3 \end{bmatrix} \tag{4.27}$$

This augmented system is only used by the error estimation mechanism (second and third equation of (4.7)), and not by the simulation of the nonlinear model (first equation of (4.7)), that instead uses only the non-augmented part of the state vector.

The entire system can finally be implemented through the simulation software Simulink, in a way that resembles the already-shown block diagram in Figure 4.1, leaving the simulation options with their default values (integration through *ode45s*, time-varying steps).

4.3 Simulation results

It is now possible to test the tool against data sets from a simulated environment. The one used in this work is described in Appendix A, along with the corresponding test trajectories.

The static parameters required by the model, m, I_z, a, and b have been filled to resemble the values from the vehicle datasheet. The starting value of the uncertain parameters can be set through the last two elements of the initial condition vector x_0, chosen to be approximately at the middle of the tool operating range

$$x_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 50000 & 50000 \end{bmatrix}^T \tag{4.28}$$

while the initial conditions of vector P can be left zero

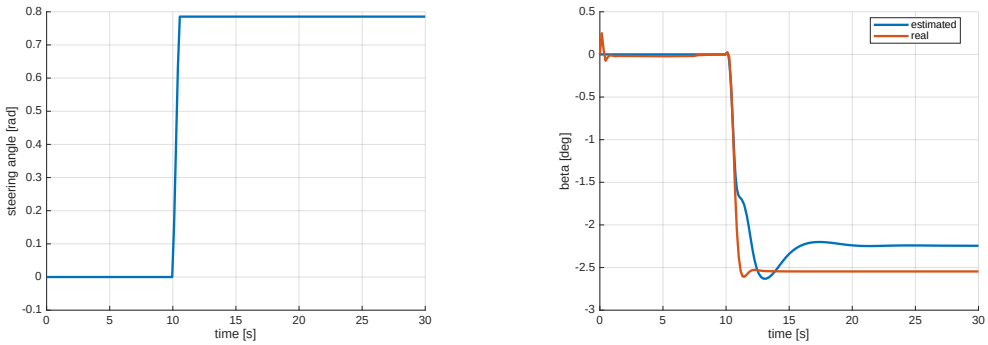
$$P_0 = [7 \times 7] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.29}$$

Roundabout trajectory

A first test can be performed against data from the virtual vehicle running at a constant longitudinal velocity of $80\ km/h$ and following the roundabout trajectory (Section A.3), with the wheel oriented at $45°$, reported again in Figure 4.2a for clarity reasons. Ideal position measurements are fed into the estimator.

Results displayed in Figure 4.2b show that the estimator is able to track the sideslip angle, but a drift error arises and remains constant, due to the uncertainties present

in the static parameters of the model. This problem also affect the other model-based estimators treated in this work: the mass, inertia and distances a and b of the single track model have a very high influence on the precision of the results, a phenomenon that will be discussed in the subsequent chapters.



(a) Steering signal **(b)** Estimation

Figure 4.2: Results of the EKF estimator against the roundabout trajectory

The evolution of the cornering stiffness parameters (the last two elements of the augmented state vector) is shown in Figure 4.3, clearly showing that the estimation stabilizes around two constant values, that corresponds to the one identified by alternative methods (the greybox procedure provided by Dymola results in $C_f \approx 57000$ and $C_r \approx 47000$).

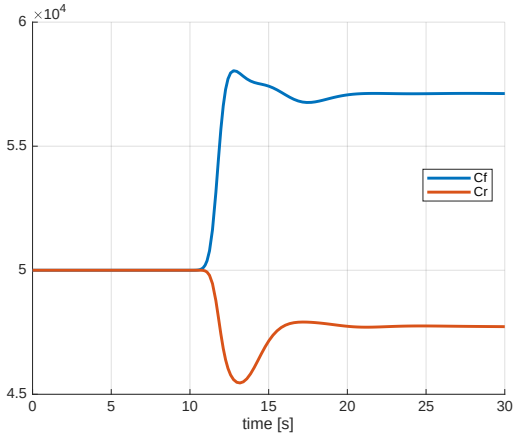
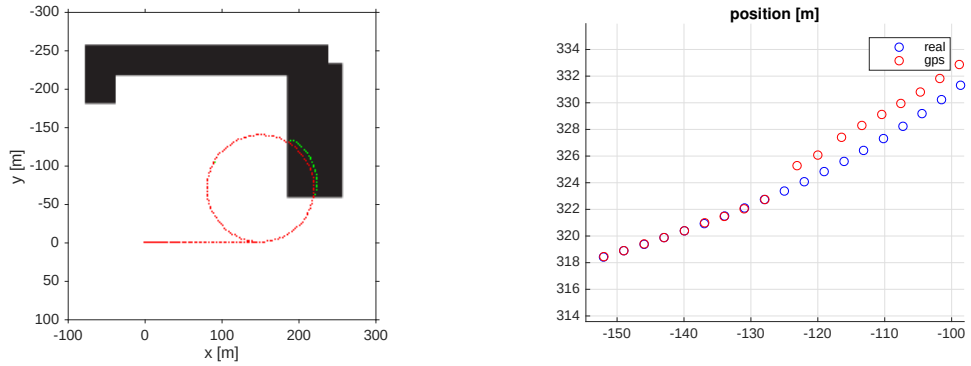


Figure 4.3: Evolution of Cf and Cr with the EKF estimator

GNSS robustness

The measurement noise robustness can be tested by running the estimator against data from the same simulation, but this time using position measurements affected

by a GNSS error model, subjected to sudden drifts, as explained in Section A.2 and represented by the noise map in Figure 4.4a. The variance and mean of the noise are set to respectively $0.02\ m$ and $5\ m$, causing sudden drifts like the one represented in Figure 4.4b.



(a) Vehicle position above noise map **(b)** Example of measurement shift

Figure 4.4: GNSS error model applied to the roundabout trajectory

The estimation is negatively affected by the GNSS noise model, and the induced errors are compensated only after some seconds, resulting in sideslip values different from the ones of the original test, as shown in Figure 4.5. This is expected, since the EKF, when used for identifying parameters, is very sensitive to measurement errors, as it does not consider the measurements over the entire timespan, but only the ones received until the considered time instant. A solution consists in decreasing the noise covariances in R and Q, but this would also affect the bandwidth of the tool, so it is not advisable.

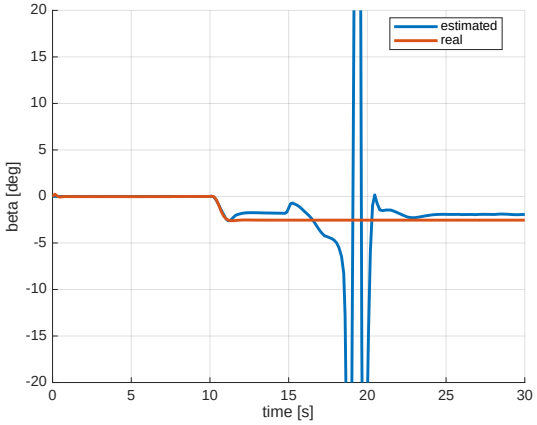
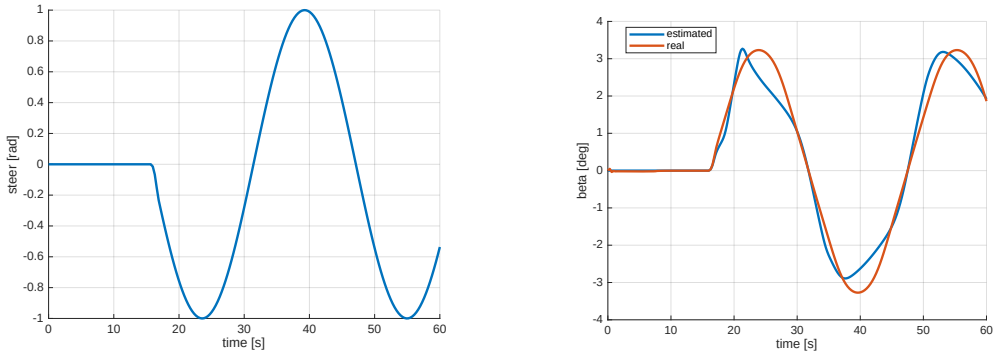


Figure 4.5: Results of the EKF estimator against the roundabout trajectory, with the GNSS error model

Slalom trajectory

Another test can be run against data from the virtual vehicle running at a constant longitudinal velocity of 80 km/h and following the slalom trajectory (Section A.3) reported in Figure 4.6a. This time the drift error is less present, as the tool continuously vary the uncertain parameters to adapt itself to the variability of the track, a clear advantage over other approaches, that use constant values of C_f and C_r over the entire timespan, resulting in worse estimates.



(a) Steering signal **(b)** Estimation

Figure 4.6: Results of the EKF estimator against the slalom trajectory

4.4 Experimental results

A round of tests can be performed against real data sets obtained from the driving of the unmanned ground vehicle described in Appendix B. Among the various experiments performed, a single data set has been chosen to show the effectiveness of the tools treated in this work and to compare the results between each other. The initial condition vector is filled with measurements corresponding to the first considered time instant, and the starting values of the uncertain parameters are lowered on the basis of previous tests

$$x_0 = \begin{bmatrix} 0 & 0 & \psi_0 & x_0 & y_0 & 25 & 25 \end{bmatrix}^T \tag{4.30}$$

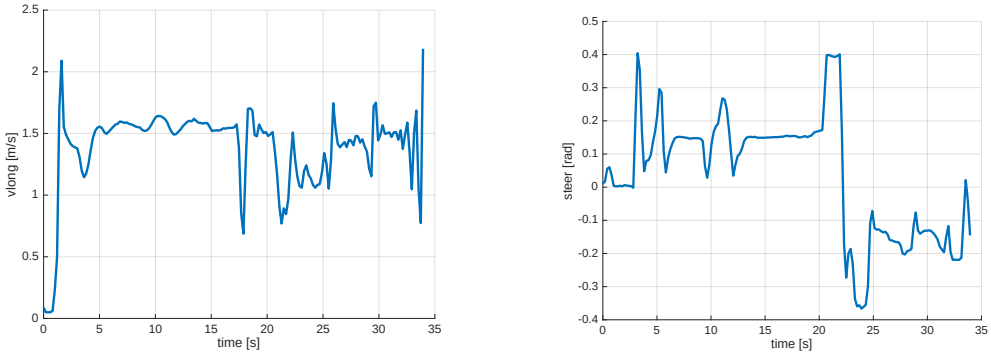
The other parameters needed by the filter are left unchanged.

The input quantities related to the test are represented in Figure 4.7. Results of the estimation of the entire state vector of the single track model, used by the tool, are reported in Figure 4.8. The values compared with the estimated quantities and used as reference are:

- For the sideslip angle, the value computed by the optitrack system through the vehicle velocity;

- For any other quantity, the value read by its dedicated sensor, i.e. the gyroscope for the yaw rate and the optitrack for the position and attitude.

An offset is clearly visible between the sideslip value given by the optitrack and the one given by the estimator, even if the latter is able to correctly interpolate the position measurements, as shown in Figure 4.8d and Figure 4.8e.



(a) Longitudinal velocity **(b)** Steering signal

Figure 4.7: UGV Inputs

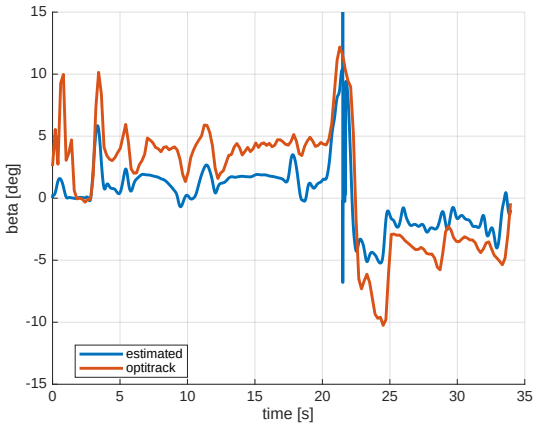
The quality of the estimate can be asserted by introducing the *Theil index*, a statistic used to evaluate the difference between a model output and the available measurements, defined as

$$TIC = \frac{\sqrt{\sum_i (y_i - y_{m,i})^2}}{\sqrt{\sum_i y_i^2} + \sqrt{\sum_i y_{m,i}^2}} \tag{4.31}$$

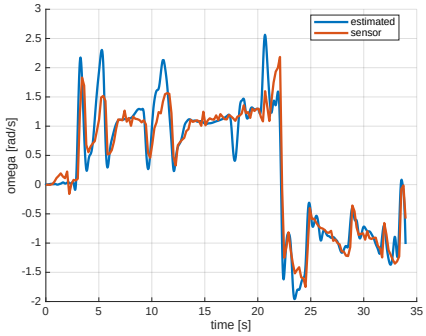
where i is the time index, y_i the output of the model at time instant t_i, $y_{m,i}$ the corresponding measurement.

The TIC parameters corresponding to the two measurements used in this test, that are the vehicle position values referred to the x and y axes, are reported in the table below.

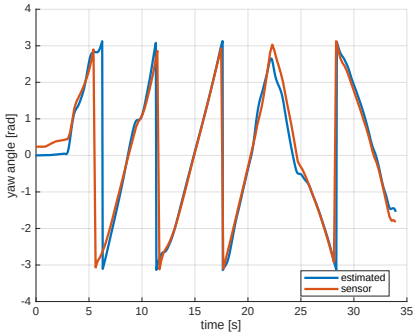
TIC_x	0.0011909
TIC_y	0.00076763



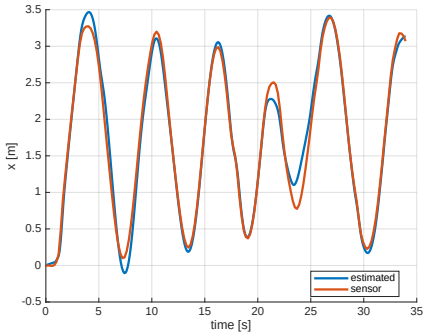
(a) Slip angle



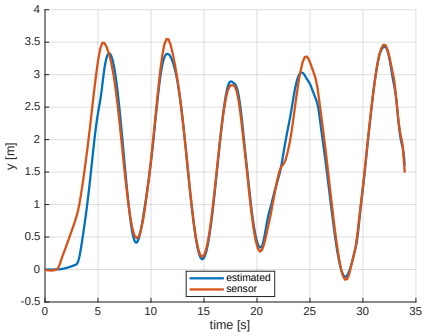
(b) Yaw rate



(c) Attitude



(d) Position (x axis)



(e) Position (y axis)

Figure 4.8: Results of the EKF estimator against data from a UGV

4.5 Conclusions

A sideslip estimator based on the Extended Kalman Filter, the de-facto standard when dealing with parameter identification, has proven to be a versatile tool that can be applied to different situations without particular difficulties, although the convergence of the estimation can not be guaranteed, as it depends on the choice of the initial conditions and on the values of the matrices Q and R.

At the same time, a series of limitations have emerged:

- The tool is highly sensible to sudden shifts in the measurements, like the one introduced by a GNSS error model, and struggles to compensate the error, sometimes resulting in wrong estimates.
- The static parameters can critically influence the result precision, introducing an irremovable bias error, an issue shared with all the vehicle dynamic model-based estimators. A solution consist in augmenting the number of uncertain parameters, at cost of introducing more sensors, but the estimation would result more difficult. Moreover, the behavior of the tool with additional uncertain parameters is all to be proven.
- The tool does not provide any mechanism to define a proper search direction, to weight the various estimation errors and to impose constraints: a solution is always searched unconditionally. This can constitute a serious limitation in case of noisy data sets, or when multiple uncertain parameters are introduced.

These conclusions will serve as a basis for the implementation and analysis of the tools presented in the next chapters.

Chapter 5

Slip angle estimation through Feedback Linearization

This chapter is dedicated to a sideslip estimator based on the feedback linearization technique. Dynamic equations are taken from the unicycle model, a simple relationship between the vehicle kinematic properties. The algorithms exploits measurements from a single position sensor (optical and GNSS are considered). The tool is presented, improved considering practical issues, and tested against available data. Finally, implementation notes are provided.

5.1 Motivations

As already mentioned in the state of the art chapter, a series of estimators are based on simple kinematic considerations, and do not make use of a vehicle dynamic model, that introduces a series of limitations due to its assumptions and uncertainties. In particular, GPS/INS-based estimators calculate the sideslip angle β by simply removing the attitude (ψ, yaw angle) from the heading (car direction, θ)

$$\beta = \theta - \psi \tag{5.1}$$

These values can be computed by feeding the measurements from one or two GPS and INS to a filtering algorithm like the Extended Kalman Filter [37]. This approach has at least two limitations:

- Extended Kalman Filter is composed by at least 3 differential equations, and requires a repeated linearization of the system to propagate the statistical properties, a procedure which, depending on the implementation, is associated to a considerable computational cost, particularly negative if our aim is providing estimations with a high bandwidth.
- β is highly influenced by the measurement accuracy, which depends on the sensor characteristics. As already explained, GNSS sensors are characterized

by a low resolution, noise and sudden shifts in presence of obstacles. The EKF is able to filter out the noise, but the effect is limited and the bandwidth is reduced.

The accuracy problem can only be solved by an upgrade of the sensors; we can instead focus on optimizing the computational cost, an operating that can be performed by replacing the filtering algorithm with a lighter procedure.

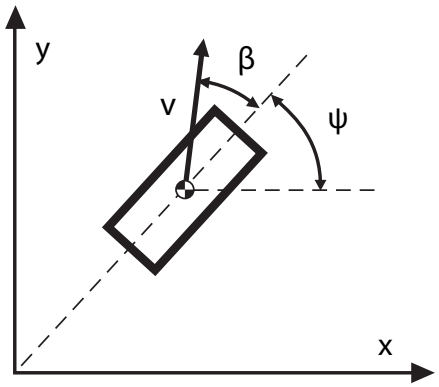


Figure 5.1: Unicycle model

To build this procedure, we have to start from a proper differential model. We consider the Unicycle model represented in Figure 5.1, a simple differential model that links a body velocity vector v and yaw rate ω with its absolute position and heading, represented in this chapter as the sum of the attitude and sideslip angle $(\psi + \beta)$

$$\begin{cases} \dot{x} = v \ \cos(\psi + \beta) \\ \dot{y} = v \ \sin(\psi + \beta) \\ \dot{\psi} + \dot{\beta} = \omega \end{cases} \tag{5.2}$$

The idea is to use the position measurements x and y to obtain values of the other two variables, v and $\theta + \beta$. x and y need an integration, so they must be considered state variables of the system and can not be inserted directly. Instead, the model must be simulated and updated with available measurements, a practice performed through an *observer*, a standard control theory tool, similar to the Kalman Filter (although built on different basis). The task is difficult due to the non linearity of the equations: unlike the linear case, implementing an observer for a nonlinear system is non trivial and requires a specific approach.

A first option consists in linearizing the system through the standard jacobian method and using the canonical observer for linear systems, but it would introduce a significant approximation with a negative influence on the results. The feedback linearization is one of the available alternatives, as it allows to build an observer-like system without approximations.

The application of feedback linearization to the unicycle has been proposed in several works and for various purposes [10] [38]. Its robustness is still to be tested, in particular with respect to the characteristics of available sensors. A robust implementation could provide an easy sideslip estimation algorithm, particularly suitable when precise positioning systems are available.

5.2 Feedback Linearization

Feedback linearization is an approach to control design that aims at transforming nonlinear dynamic models into equivalent linear models, which can be used with standard control techniques, such as PID, LQR or Pole Placement. It requires a time invariant, nonlinear model that can be written in this standard form

$$\begin{cases} \dot{x} = f(x) + g(x)u \\ y = h(x) \end{cases} \tag{5.3}$$

where $x(t)$ is a n-dimensional state vector and $u(t)$ a m-dimensional vector of input variables. The method consists in finding an arbitrary transformation of the state variables $z = T(x)$ that allows to build a new, linear system, equivalent to the original one [21], such as

$$\begin{cases} \dot{z} = Az + Bv \\ w = Cz \end{cases} \tag{5.4}$$

where z is an r-dimensional vector of transformed state variables, v is a m-dimensional vector of transformed input variables, w is a m-dimensional vector of transformed output variables and A,B,C are constant matrices.

The nonlinear part, removed through the transformation, is moved in a control law, that precedes the original system. In some cases, linearization can be achieved with a static control law, while in others a nonlinear dynamic state feedback law (compensator) is required:

$$\begin{cases} \dot{\xi} = a(x, \xi) + b(x, \xi)v \\ u = c(x, \xi), + d(z, \xi)v \end{cases} \tag{5.5}$$

where ξ is a q-dimensional vector of compensator state variables.

In practice, an additional closed loop is introduced to actively update the additional control law, as shown in Figure 5.2. The overall loop can be considered as the one of a completely linear plant, but without the approximations introduced by the standard Jacobian linearization.

The transformed system is usually equivalent to a set of decoupled input-output chains of integrators, and so can be simulated with standard numerical methods.

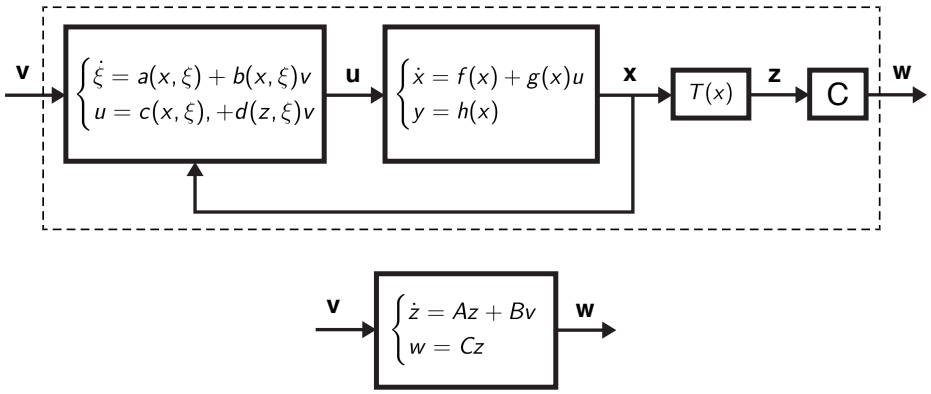


Figure 5.2: Feedback linearization

5.3 Application

Linear dynamic models can be used to estimate or update state variables through the implementation of observers, virtual systems fed by available state variables and control actions, that predict output values and use these predictions to correct themselves through a closed loop. An observer based on a linear system has this typical form, also represented in block diagram form in Figure 5.3a

$$\begin{cases} \dot{\hat{x}} = A\hat{x} + Bu + K[y - \hat{y}] \\ \hat{y} = H\hat{x} \end{cases} \tag{5.6}$$

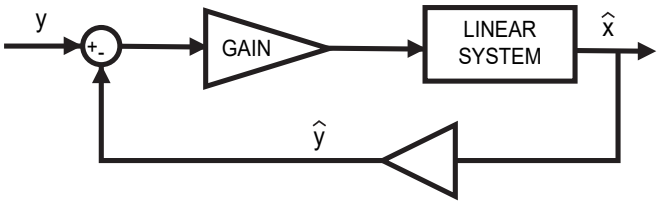
Where quantities denoted with the hat are estimations, while the others are available quantities, fed into the system at every iteration. Error convergence is ensured by the term $K[y - \hat{y}]$, whose gain K dictates the bandwidth.

Feedback linearization should allow to exploit this approach with any nonlinear system, by applying the previously described transformation. A possible implementation is shown in Figure 5.3b: the system has been replaced with an equivalent linear system, preceded by a control law.

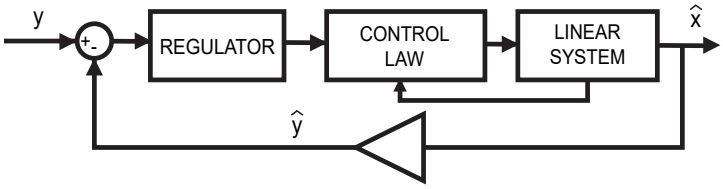
The gain has also been replaced by a generic regulator, that will be defined according to requirements.

We can now go back to the unicycle model defined in equation (5.2). We are interested in estimating the state variables x and y: since we know that the result will be a chain of integrators, some variables able to control independently the position values are their accelerations. The equations corresponding to x and y are derivated, obtaining

$$\begin{cases} \ddot{x} = \dot{v}\cos(\psi + \beta) - v(\dot{\psi} + \dot{\beta})\sin(\psi + \beta) \\ \ddot{y} = \dot{v}\sin(\psi + \beta) + v(\dot{\psi} + \dot{\beta})\cos(\psi + \beta) \end{cases} \tag{5.7}$$



(a) Normal observer



(b) Feedback linearization

Figure 5.3: Feedback-linearized observer compared to a normal observer

We consider the nonlinear law developed for the tracking of the unicycle [27]

$$\begin{cases} \dot{\xi} = u_1 \cos(\psi + \beta) + u_2 \sin(\psi + \beta) \\ v = \xi \\ (\dot{\psi} + \dot{\beta}) = \dfrac{u_2 \cos(\psi + \beta) - u_1 \sin(\psi + \beta)}{\xi} \end{cases} \tag{5.8}$$

and replace v and $(\dot{\psi} + \dot{\beta})$ in the previous equations

$$\ddot{x} = u_1 \cos^2(\psi + \beta) + u_2 \sin(\psi + \beta)\cos(\psi + \beta) + \\ - u_2 \cos(\psi + \beta)\sin(\psi + \beta) + u_1 \sin^2(\psi + \beta) \tag{5.9}$$

$$\ddot{y} = u_1 \cos(\psi + \beta)\sin(\psi + \beta) + u_2 \sin^2(\psi + \beta) + \\ + u_2 \cos^2(\psi + \beta) - u_1 \sin(\psi + \beta)\cos(\psi + \beta) \tag{5.10}$$

from which it is clear that x and y are independently linked to the inputs u_1 and u_2, and controllable through them

$$\begin{cases} \ddot{x} = u_1 \\ \ddot{y} = u_2 \end{cases} \tag{5.11}$$

It is important to emphatize the fact that the control law has a singularity in $\xi = v = 0$, that verifies when the unicycle is not rolling, a common limitation already anticipated in the vehicle dynamics chapter. Despite the fact that a sideslip

estimator for $v \to 0$ is useless, this issue must be kept into account when implementing the controller [27], for instance by choosing an adeguate initial value for ξ and by resetting it when a threshold is passed. The transformed model results

$$\begin{cases} \dot{z}_1 = \ddot{\hat{x}} = u_1 \\ \dot{z}_2 = \ddot{\hat{y}} = u_2 \end{cases} \tag{5.12}$$

This represents a chain of two input-output integrators, in which it is evident that x and y are decoupled and controllable separately by u_1 and u_2.

We can now transform the system into an estimator, by ensuring the error convergence between the state variables and the measurements, using a regulator. We select a standard PID control, without the integral action, since the system itself is already equivalent to a double integrator. The control actions are fed into the new inputs of the feedback-linearized system:

$$u_1 = k_p(x - \hat{x}) + k_d(\dot{x} - \dot{\hat{x}}) \tag{5.13}$$

$$u_2 = k_p(y - \hat{y}) + k_d(\dot{y} - \dot{\hat{y}}) \tag{5.14}$$

k_p and k_d can be tuned to achieve a desired bandwidth. The resulting tool, shown in Figure 5.4, is undoubtedly lighter than Extended Kalman filter and similar techniques; moreover, the convergence between the measurements and the position components can be controlled more easily, thanks to the dedicated PD regulators, while in the EKF it is generally difficult to tune a single state variable, although a precise statistics propagation method is present. Of course, a precise performance comparison can be performed, and is left to a future work.

As already anticipated, the robustness of the proposed estimator against available sensors is all to be tested, and can only be determined experimentally.

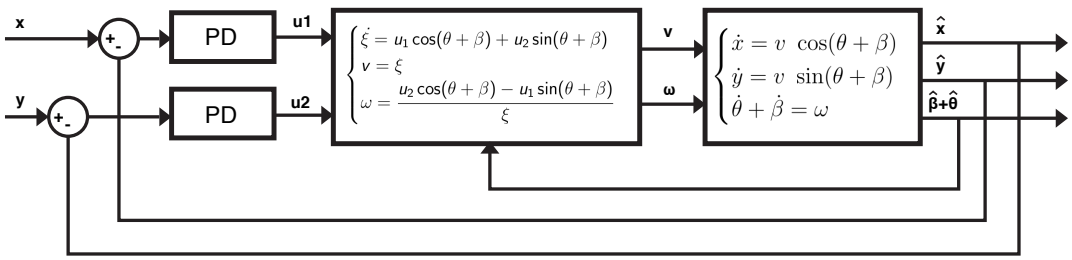


Figure 5.4: Feedback linearization-based sideslip estimator

The sideslip estimator can be implemented in Simulink through the use of blocks, resembling Figure 5.4. The equations are integrated with the default *ode45s* estimator and a varying time step.

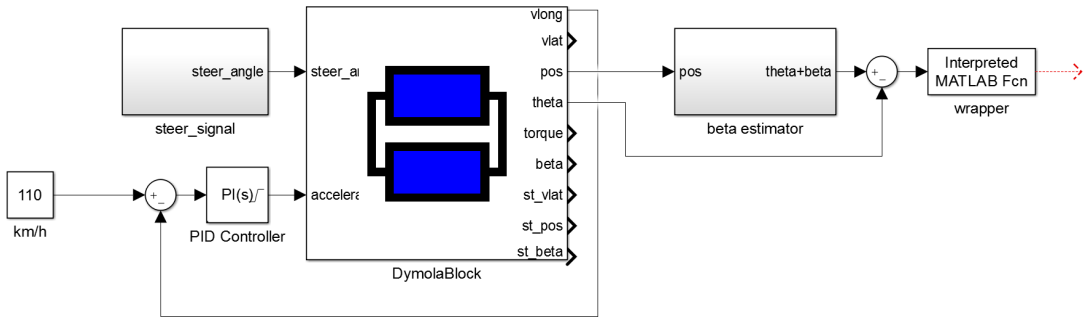


Figure 5.5: Setup of the complete FL-based estimator

The only parameters that have to be tuned are the PD gains of the estimators, that dictate the internal loop bandwidth and is usually limited by stability issues, not present in this case. The choice is limited only by the need of obtaining a reasonable simulation speed, since an high bandwidth causes smaller integration time steps. Typical gains can be

$$k_p = 500 \tag{5.15}$$
$$k_d = 100 \tag{5.16}$$
$$N = 100 \tag{5.17}$$

Since the control is a PD, equivalent to a zero, an additional pole has to be introduced to restore causality, controlled by the N parameter. The slip angle β is finally obtained by subtracting the vehicle attitude ψ from the estimator output $(\psi + \beta)$. The result can be compared to the value measured by an ideal sensor. The final Simulink setup is shown in Figure 5.5.

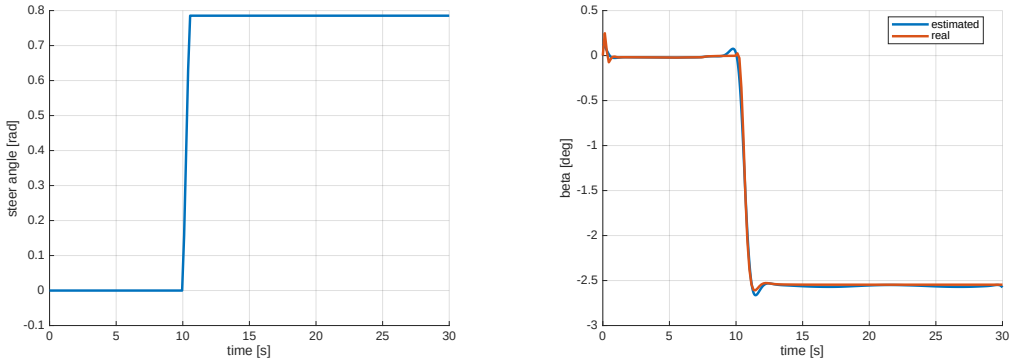
5.4 Simulation results

The tool has been tested against data sets from the simulated environment used throughout the entire work, whose setup and characteristics are reported in Appendix A.

Roundabout trajectory

A first test can be performed against data from the simulation of a vehicle running at a constant longitudinal velocity of 80 km/h and following the roundabout trajectory (Section A.3), with the wheel oriented at 45°, shown in Figure 5.6a, using the ideal position measurements as the estimator input. Results, reported in Figure 5.6b, show that the proposed estimator is perfectly able to follow the behavior of the car model implemented in Dymola, with a negligible steady-state error.

Eventual oscillations can be mitigated by a better tuning of the PID parameters of the loop.



(a) Steer signal **(b)** Estimation

Figure 5.6: Results of the FL estimator against the roundabout trajectory

GNSS robustness

We proceed to introduce the GNSS error model on the position measurements, as described in Section A.2 and in the same way as performed with the previous estimator. The noise map is the same as Figure 4.4, to allow a precise comparison.

Results of the new simulation are shown in Figure 5.7. We can observe a measurement noise effect, and a series of sudden jumps in the estimation, corresponding to a change of GNSS quality, that however are immediately compensated. The stability seems not affected by sudden shifts in the position measurements, and is apparently very robust against disturbances, a major improvement with respect to the EKF-based estimator described in Chapter 4.

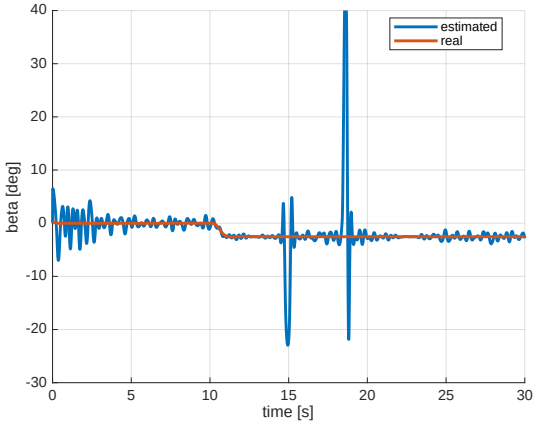


Figure 5.7: Results of the FL estimator against the roundabout trajectory, with the GNSS error model

The noise, both in terms of variance and shifts, can be filtered out. Various tests have concluded that it is better not to edit the PID gains for this scope, as lower PIDS cause a decrease of bandwidth, with a slower error compensation. An alternative and flexible solution consists in using two blocks of low-pass filters, one placed on the estimator input, and the other on the estimator output, as shown in Figure 5.8: various combinations of tuning allow the estimation of proper values for almost any analyzed situation.

Figure 5.8: Filtered FL estimator

Angle normalization

By increasing the estimator bandwidth or using data sets from real tests, it is possible to incur in the situation shown in Figure 5.9: while the error converges to zero, β stabilizes around increasing values.

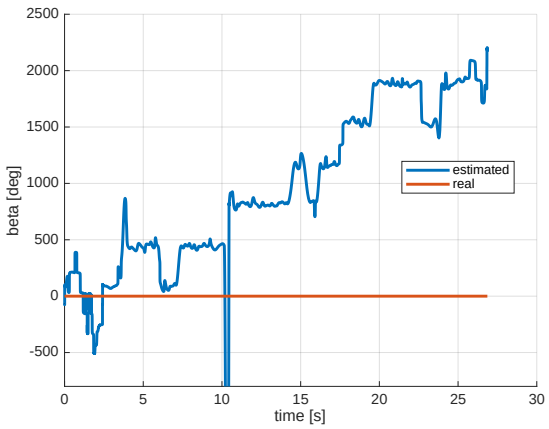


Figure 5.9: Simulation without angle wrapping

Repeated testing allows to assert that this problem depends on two distinct phenomena:

- The heading, subtracted from the estimator output ($\psi + \beta$), could not wrapped in a reasonable range.

- The estimator gains are so high that β stabilizes around progressively higher values, since it passes through trigonometric functions and an integration that do not care about its real value.

A solution for both problems consists in wrapping the estimator output value in the range $(-180°; +180°]$. A decrease of the PID gains is also possible, but with the drawback of a less responsiveness to errors.

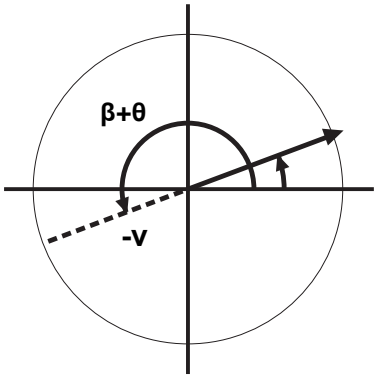
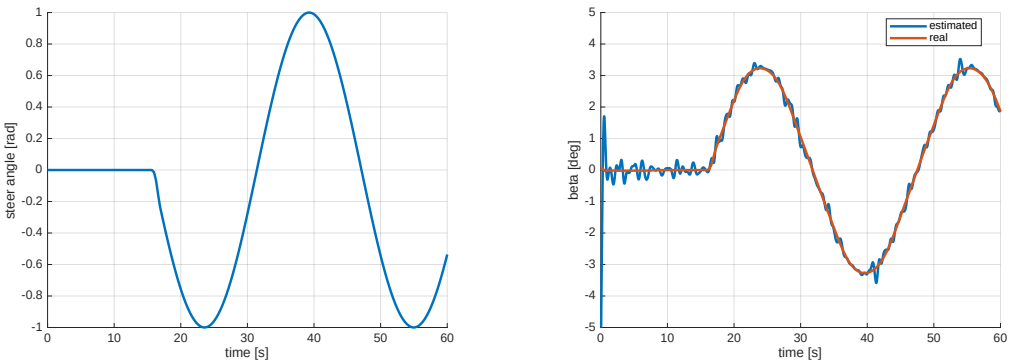


Figure 5.10: Beta meaning in case of negative velocity state values

Another problem arises when the velocity state variable of the estimator becomes negative, a problem caused by the tool bandwidth or by excessively disturbed data. In this case, to get a proper β, it is necessary to detect the phenomenon and add $\pm 180°$, according to the beta sign, as can be deducted from Figure 5.10.

Slalom trajectory



(a) Steer signal **(b)** Estimation

Figure 5.11: Results of the FL estimator against the slalom trajectory

The complete tool can be tested against data from the vehicle running at 80 km/h and following the slalom trajectory (Section A.3). Since the robustness of the tool against measurement shifts has already been demostrated and addressed, the GNSS model parameters (variance and shift) have been drastically increased to test the reliability of the final version of the tool. The estimation, reported in Figure 5.11b, is apparently not influenced by longitudinal velocity, road adherence, variability of forces and other factors that constitute a problem in parameter identification-based estimators.

Drifting simulation

An additional test has been performed by using data from the simulation of a single track model in a complete non-adherence situation, called drifting, stabilized through the pole placement technique. In such scenario, a precise sideslip identification cannot be performed by most estimators that use the single track model with a constant cornering stiffness, as F_y and C_f become completely unrelated due to saturation. Since the considered method does not use a lateral dynamics model, the situation can be handled, as clearly shown in Figure 5.12. The oscillation at the end of the plot is a glich caused by the time-varying simulation, also present in other tests against data sets from "perfect" simulations and without noise.

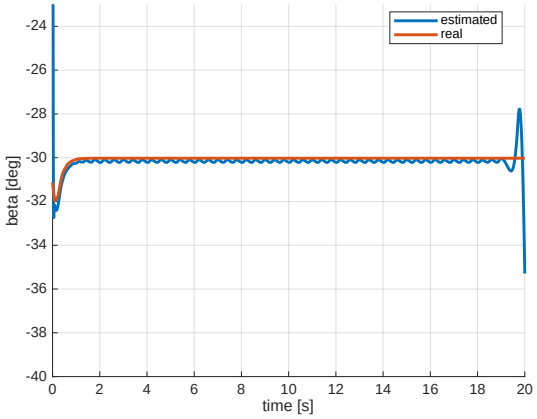


Figure 5.12: Results of the FL estimator against a drifting simulation

5.5 Experimental Results

An estimation has been performed against a data set from the driving of the Unmanned Ground Vehicle described in Appendix B, used throughout the work. The

low-pass filters introduced during the simulation phase have been implemented as second-order low pass filters, with a damping factor of 1.5.

The slip angle plot of Figure 5.13 shows that the values provided by the tool and the one provided by the optitrack system, used as reference, are characterized by a very little relative error. This is expected, since both are based on the same measurements (the vehicle position), and do not perform any merging with data from other sensors. This constitutes a limitation, since any possible measurement error can not be compensated. Nevertheless, considering the simplicity of the tool, it is safe to assume that realistic estimates can be provided after a proper calibration.

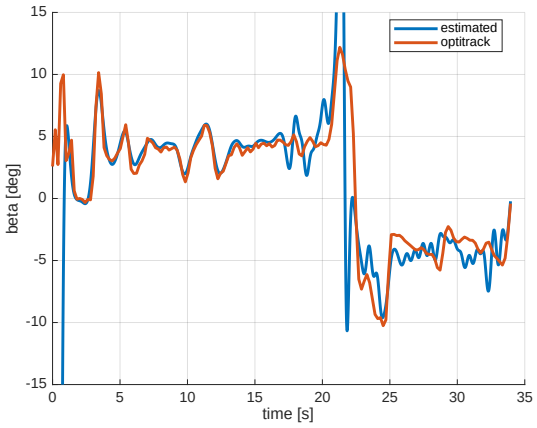


Figure 5.13: Results of the FL estimator against data from a UGV

5.6 Implementation notes

Since the estimator is meant to be used as a supporting on-board tool, the algorithms have been successfully implemented for testing purposes in a Robot Operating System (ROS) node, to be used in a ROS-running machine. ROS is not a real operating system nor a hard real time system, but a set of programs and libraries that provide a strong underlying platform for the development and interaction of control system components.

The writing of such estimator in a programming language presents more or less the same problematics, independently from the platforms, so the one encountered for the specific case are reported in this section.

The core of the estimator is a set of first-order Ordinary Differential Equations (ODEs), that must be solved through a numerical method, that have to be chosen and implemented, or called through available libraries. The choice of a simple integration algorithm with a deterministic execution time can nevertheless lead to decent results, in particular if bandwidth is a constraint or there are no libraries available, like in the case of a microcontroller.

Another task is the realization of the PD regulator: it can be directly expressed in time domain or realized from Laplace domain through a proper transformation (Tustin, explicit Euler, implicit Euler).

As introduced in multiple points, the estimator has a singularity for $v \to 0$: before computing the control law of the feedback linearization, the implementation must verify that $abs(v) > tollerance$, otherwise the tool must enter in a reset mode, where integrated variables are set back to their initial value, and internal position is imposed as the current position.

An important aspect is related to the step synchronization with respect to the input values:

- The loop containing all the solving procedures can be run synchronously with respect to incoming data from sensors: in this way the implementation is simpler, but output frequency depends on the one of the input data. Moreover, big time intervals between samples could influence the estimator stability, that must be kept under control.
- The loop can otherwise be run asynchronously, using the zero order hold method and outputting values at constant rate. Aliasing must be avoided, and implementation is more difficult, since two tasks (data gathering and model simulation) must be run concurrently, making necessary the use of methods for parallelization, like asynchronous programming or threads.

5.7 Conclusions

The proposed approach is much cheaper than standard solutions both in terms of economic cost and required computational power; it works well when a uniform GNSS signal is available, can work with minor adjustments when the signal level varies. It is also relatively simple to implement on any possible platform, as the underlying math is limited. The system is also the most reliable when dealing with large slip angle values, as it does not depend on a vehicle dynamic model and can handle better a change of behavior, at cost of a greater dependency on the measurements accuracy.

The cons are a limited bandwidth and robustness, since they are both dictated by sensor characteristics. Furthermore, control parameters must be tuned manually, according to the current scenario.

Since the only input is constituted by positioning measurements, the possibilities of improvements are limited:

- The positioning sensor can be replaced with a more precise system.
- A more precise positioning system could be obtained by feeding inputs from both a position sensor and a IMU sensor into an Extended Kalman Filter, increasing the bandwidth.

Chapter 6

Base slip angle estimation through LFT

This chapter is dedicated to a sideslip estimator based on the single track model, a vehicle lateral dynamics representation, and on a parameter identification technique for nonlinear models developed by exploiting a Linear Fractional Transformation (LFT) of the model, in order to minimize the error between the predicted output and available measurements. This technique, highly flexible, is explained starting from the static minimization problem and applied to the problem through an existing toolbox, improved with the introduction of new functionalities. An implementation of the system is then tested against available data.

6.1 Motivations

While a sideslip estimator based exclusively on kinematic considerations, like the one based on the feedback-linearized unicycle, is characterized by a good flexibility (after a proper tuning), a series of limitations are also evident: robustness is highly dependent on the measurements accuracy, that can change over time, and bandwidth is limited by the frequency at which sensor values are provided. This can constitute a problem if we need a higher bandwidth, for instance if the estimation is intended for a safety system, or if we need to perform predictions, i.e. estimating beta values in the future, according to a certain steering angle and velocity reference.

Model-based estimators are intended to address these limitations, as they provide the needed values through a simulation of a vehicle dynamic model. The drawback is a reduced operating range, since the models have inherent uncertainties and can rarely be reliable without the introduction of limiting assumptions.

The most used model in sideslip estimation is the already introduced single track (equations (3.20)), sometimes coupled with a tyre model (Pacejka, Burckhardt or Kiencke and Daiss). Its differential equations include at least 6 parameters (a, b,

m, I_z, C_f, C_r), that must be chosen to replicate the current vehicle behavior. While the vehicle geometric properties a and b (distance of COG from front and rear wheel axle) and also inertial properties m and I_z can be read from datasheets or estimated through common methods, the cornering stiffnesses C_f and C_r depend on the road and vehicle specific conditions, and must be identified during the vehicle motion, by placing an algorithm on a processing unit in the car, fed by available sensors. This constitutes an additional task, harder than usual due to the fact that the model is nonlinear, so a large part of the control and identification theory approaches cannot be applied.

The most used tool to perform this kind of work is undoubtedly the Extended Kalman Filter, not only because of its reliability, but also because the parameter identification procedure is performed in the same step of the model output computation, resulting in a simple tool, as shown in Chapter 4. However, a series of limitation are present:

- the model has to be continuously linearized, an operation with an high computational cost;
- the parameter identification procedure, that consists in augmenting the system with an additional equation that calculates the error due to the parameter value, is slow and difficult to control, since its convergence is controlled by the covariance matrix P, that is intended for dictating the covariance of the state values;
- the convergence of the parameter identification procedure strongly depends on the initial conditions of the model.

Different approaches can be developed, especially if the identification problem and the estimation problem are split up. We could, for instance, split the problem in two separate parts:

1. Identify the uncertain parameters of the model, by minimizing the error between the model output and available measurements, obtained in a certain timespan.
2. Compute the output of the model, using the identified parameters, until a new identification is necessary. The identification can be performed before or during the output computation, in a parallel process.

We focus on the identification problem, that can be expressed as a *static minimization problem*, intended as a problem in which the error between available measurements and model output is minimized, and the operation is performed once for each considered timespan.

This operation is particularly difficult for nonlinear systems, in terms of algorithms and computational costs. The available numerical algorithm to solve the minimization problem use extensively gradient matrices, Hessian matrices and Gauss-Netwon directions, that are typically obtained through an inefficient linearization of the system. Moreover, the matrices are often inverted, a procedure

with an high computational cost (it is the bottleneck of the EKF [39]) and subjected to errors in case of high condition numbers.

Various alternative approaches have been developed, based on prediction Error Method (PEM) [25], two-step method (TS) [17], backtracking search algorithms [12], white box, grey box [26] and many more.

In this work, the analyzed parameter identification approach is based on the Linear Fractional Transformation (LFT) formulation of nonlinear systems, a common formalism used both in control and identification, that separates the linear, nonlinear and uncertain part of the system. A model written in LFT form can be passed to a properly-built algorithm, presented in the next section, that computes the gradient and Hessian at each time instant, efficiently and without any approximation. Then, the chosen numerical algorithm can solve the minimization problem, obtaining the most suitable parameters.

This technique allows to identify the missing parameters of the single track model, that can then be simulated at arbitrary time instants, estimating β, even in the future (predictor). The process is highly flexible, as the choice of the parameters to identify, the available measurements used for the minimization and the frequency of the identification is left to the user.

In the following sections, the two-stage LFT algorithm is explained and applied to the single track model in its base form.

6.2 Two-stage LFT estimator

6.2.1 Parameter identification problem

We consider a discrete nonlinear model

$$\begin{cases} x(k+1) = f(x(k), u(k), d) \\ y(k) = h(x(k), u(t), d) \end{cases} \tag{6.1}$$

where f and h are function vectors containing the model equations, d is a vector containing uncertain parameters, $x(k)$ is the model state vector at time instant t_k, $u(k)$ the input vector, $y(k)$ the output vector.

The parameter identification problem can be formulated as a minimization problem (least squares problem), with the following syntax [7]: *given inputs* $u(k)$ *and output measurements* $y(k)$ *at sampling instants* t_k*, with* $k = 1, ..., N$*, find the values* $\hat{\delta}$ *that minimizes the quadratic cost function*

$$J(\delta) = \frac{1}{2N} \sum_{k=1}^{N} e(k, \delta)^T e(k, \delta) \tag{6.2}$$

Where e at a certain time instant t_k is the error between the predicted output and a set of chosen measurements, that must have a corresponding variable in the model output vector $\hat{y}$

$$e(k, \delta) = y(k) - \hat{y}(k, \delta) \tag{6.3}$$

Neglecting the case in which J is trivial and an algebraic solution exists, the problem can be solved through one of the available numerical methods for minimization, run in a loop until the error drops below a certain tolerance. The most common are listed below.

Simple iterative search

The minimum value of J can be simply searched iteratively for the smallest value, avoiding the computation of the gradient or Hessian:

$$J_{n+1} = J_n + \Delta J_{n+1}(\delta_n + \Delta\delta_{n+1}) \tag{6.4}$$

where n is the step index and Δ is an arbitrary-chosen step value. This method is generally inefficient.

Gauss-Newton algorithm

This algorithm is a modification of the Newton method (a root-finding algorithm), applied to the minimization field, that introduces the gradient and Hessian into the problem. A necessary condition for optimality is that the gradient of the cost function must be zero with respect to the optimizing parameters [39]

$$\frac{\partial J}{\partial \delta} = 0 \tag{6.5}$$

The considered cost function can be approximated around a nominal unknown parameters vector δ_0 through the Taylor expansion, truncated at the second order term

$$J(\delta) \approx J(\delta_0) + \left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_0}\right)^T \left(\delta - \delta_0\right) + \frac{1}{2}\left(\delta - \delta_0\right)^T \frac{\partial^2 J}{\partial \delta^2}\Big|_{\delta=\delta_0} \left(\delta - \delta_0\right) \tag{6.6}$$

The necessary optimality condition results

$$\frac{\partial J}{\partial \delta} = \left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_0}\right)^T + \left(\delta - \delta_0\right)^T \frac{\partial^2 J}{\partial \delta^2}\Big|_{\delta=\delta_0} = 0 \tag{6.7}$$

By equating with zero and solving with respect to δ, we obtain

$$\delta = \delta_0 - \left(\frac{\partial^2 J}{\partial \delta^2}\Big|_{\delta=\delta_0}\right)^{-1} \left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_0}\right)^T \tag{6.8}$$

This can be written in recursive formulation and smoothed with a step size α, obtaining the final formula

$$\delta_{n+1} = \delta_n - \alpha_n \left(\frac{\partial^2 J}{\partial \delta^2}\Big|_{\delta=\delta_n} \right)^{-1} \left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_n} \right)^T \tag{6.9}$$

where n is the step index.

Since we are using a quadratic cost function (equation (6.2)), the terms in brackets can be simplified:

- The first term in brackets is the Hessian vector $H(J)$, which can be expanded as follows, neglecting the second order term

$$H(J) = \frac{\partial^2 J}{\partial \delta^2}\Big|_{\delta=\delta_n} \approx \frac{1}{N} \sum_{k=1}^{N} \left(\frac{\partial e}{\partial \delta}\Big|_{\delta=\delta_n} \right)^T \frac{\partial e}{\partial \delta}\Big|_{\delta=\delta_n} = \\ = \frac{1}{N} \sum_{k=1}^{N} E^T(k, \delta_n) E(k, \delta_n) \tag{6.10}$$

  where E is the Jacobian of the prediction error e, at time instant t_k, with respect to the unknown parameters vector δ

$$E(k, \delta) = \frac{\partial e}{\partial \delta} = - \left[\frac{\partial \hat{y}}{\partial \delta_1} \quad \frac{\partial \hat{y}}{\partial \delta_2} \quad \dots \quad \frac{\partial \hat{y}}{\partial \delta_q} \right] \tag{6.11}$$

  we recall that the prediction error is given by

$$e(k, \delta) = y(k) - \hat{y}(k, \delta) \tag{6.12}$$

- The second term in brackets is the gradient vector ∇J, which can be expanded as

$$\nabla J = \frac{\partial J}{\partial \delta} = \frac{1}{N} \sum_{k=1}^{N} \left(\frac{\partial e}{\partial \delta}\Big|_{\delta=\delta_n} \right)^T e\Big(t, \delta\Big) = \\ = \frac{1}{N} \sum_{k=1}^{N} E^T(k, \delta_n) e(k, \delta_n) \tag{6.13}$$

In case of non-quadratic cost functions, the derivatives are calculated by linearizing the cost function (Newton-Raphson iteration).

Steepest-descent algorithm

This method avoids the computation of the Hessian matrix, that is replaced with a scalar constant ϵ (gain)

$$\delta_{n+1} = \delta_n - \epsilon\left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_n}\right)^T \tag{6.14}$$

ϵ dictates the convergence bandwidth: if it is too small, convergence is slow, if too large, the convergence oscillates or may fail. The best solution is using an adaptive gain, with the following adjustment rule:

1. J_1 is calculated by using a first try ϵ;
2. J_2 is calculated by using $2 \cdot \epsilon$;
3. if $J_1 > J_2$, there's a quadratic fit through 3 points to the optimal ϵ;
4. otherwise, J_3 is calculated with $4 \cdot \epsilon$.

Quasi-Newton methods

This family of methods does not require the computation of the Hessian, and due to their flexibility they are available in most of scientific software and libraries (including Matlab). Cost function is expanded with Taylor and truncated at second order, like in the Newton Method. A quasi-Newton method replaces the Jacobian and Hessian in the equation with approximations in the neighborhood of the considered optimization point

$$J(\delta) \approx J(\delta_0) + \left(\frac{\partial J}{\partial \delta}\Big|_{\delta=\delta_0}\right)^T \left(\delta - \delta_0\right) + \frac{1}{2}\left(\delta - \delta_0\right)^T B \left(\delta - \delta_0\right) \tag{6.15}$$

where δ_0 is the optimization point and B is an approximation of the Hessian matrix, while the Jacobian results

$$\frac{\partial J}{\partial \delta} = \frac{\partial J}{\partial \delta}\Big|_{\delta_0} + B\left(\delta - \delta_0\right) \tag{6.16}$$

Setting the Jacobian to zero provides the optimality condition

$$\delta = \delta_0 - B^{-1}\frac{\partial J}{\partial \delta}\Big|_{\delta_0} \tag{6.17}$$

B is chosen to satisfy formula (6.16) with progressive approximations, through a so-called secant equation. Various formulas have been developed to numerically solve the secant equation, some requiring a quadratic cost function, like the Davidson, Fletcher, and Powell (DPF) algorithm, or Broyden, Fletcher, Goldfarb, and Shanno (BFGS) algorithm [9].

Trust-region methods

A variant of this family of methods is the one provided by Matlab and used in the LFT toolbox that will be proposed in the following sections. The problem of minimizing a quadratic cost function is well known, and, as already seen, leads to known solutions. We consider a generic cost function, this can be approximated as a quadratic form in a certain region

$$q_k(\delta) \approx J(x_k) + \nabla J(x_k)^T \Delta\delta + \frac{1}{2}\Delta\delta^T B_k \Delta\delta \tag{6.18}$$

where δ is the optimizing vector, J is the cost function, ∇J the Jacobian, B a Hessian approximation.

The quadratic form can then be tested for correctness, by computing the ratio between the predicted value corresponding to a chosen step, and the same operation performed with the original cost function

$$\rho_k = \frac{J(x_k) - J(x_{k+1}, \delta_k)}{q_k(0) - q_k((\delta_k)} \tag{6.19}$$

If the ratio is close to 1, then the quadratic form is a good approximation, and the region can be expanded, otherwise it is reduced.

From the discussed methods, we can deduce that the Hessian and gradient of the cost function play an important role in most of the numerical methods for minimization, but their computation is problematic, especially if the cost function is nonlinear, and is often avoided by replacing them with approximations.

6.2.2 LFT Formulation

The proposed LFT-based tool, developed for the identification problem, tries to provide both Hessian and gradient with a low computational cost and without approximations. The values are then passed to an arbitrary minimization algorithm, allowing the use of parameter identification in fields previously unavailable due to bandwidth constraints.

The Linear Fractional Transformation (LFT) formulation of a nonlinear differential model is a widely used formalism, for both identification and control purposes [13], in which the linear part, the nonlinear part and the uncertainties of the model are kept separated. Given a generic, differential model with uncertainty vector δ

$$\begin{cases} \dot{x}(t) = f(x(t), u(t), \delta) \\ y(t) = h(x(t), u(t), \delta) \end{cases} \tag{6.20}$$

This can be rewritten, without approximations, in the following system, also represented in Figure 6.1

$$\begin{cases} \dot{x}(t) = Ax(t) + B_1 w(t) + B_2 \zeta(t) + B_3 u(t) \\ z(t) = C_1 x(t) + D_{11} w(t) + D_{12} \zeta(t) + D_{13} u(t) \\ \omega(t) = C_2 x(t) + D_{21} w(t) + D_{22} \zeta(t) + D_{23} u(t) \\ y(t) = C_3 x(t) + D_{31} w(t) + D_{32} \zeta(t) + D_{33} u(t) \\ w(t) = \Delta z(t) = diag\{\delta_1 I_{r_1}, ..., \delta_q I_{r_q}\} z(t) \\ \zeta(t) = \Theta(\omega(t)) \end{cases} \tag{6.21}$$

where A; B; C; D; are 16 constant matrices, whose size depends on the original model. We can distinguish 3 parts:

- A linear model (first equation and 4th equation), with x, y and u as the state, output and input vectors respectively. During a system simulation, vector $\dot{x}$ is integrated to obtain x.
- A nonlinear model (last equation), composed by the function vector $\Theta(\omega)$, whose argument is the auxiliary vector ω, which is determined by the 3rd equation. During a system simulation, $\Theta(\omega)$ evaluated to determine the value of vector ζ.
- A uncertain part (5th equation), where $\Delta = diag(\delta_1, ..., \delta_q)$ is the unknown parameters vector. This is accessible through the auxiliary vector w, multiplied by another auxiliary vector, z, that is determined using the 2nd equation. During a system simulation, vector Δ is multiplied by z, returning the results in w.

It is important to empathize the fact that the simulation of a LFT formulation of a given system is formally equivalent to the simulation of the original system.

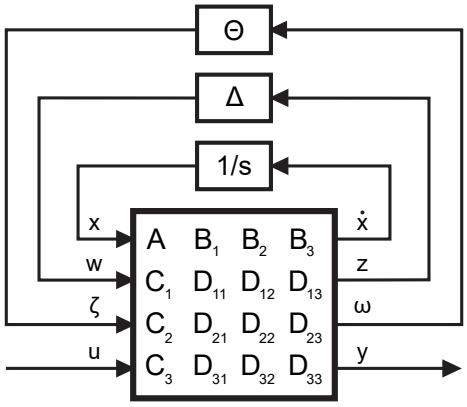


Figure 6.1: LFT formulation

This formalism was developed for a generic use, as the separation of parts can be exploited for multiple purposes; its usage for Hessian and gradient computation

has been the subject of many researches [22], and different methods have been proposed.

6.2.3 Writing procedure

Deriving the LFT formulation of a nonlinear system is a non-trivial task [7], mainly because there is not a unique solution: variables can be arranged in many ways, as there are two auxiliary vectors w and z that can be determined arbitrarily, and it is not necessary to move all the linear parts of the system into the linear equations, as they can stay in the nonlinear vector Θ at cost of a lower computational speed. Furthermore, unknown parameters can be present in parts of the equations difficult to disassemble, a situation that could require an extended reasoning. Nonetheless, a series of methods have been developed, both manual and automatic (working with a symbolic formulation of the system).

The method described below is a self-developed algorithm, that can be manually applied to different systems, including the vehicle dynamic model later used in this work, and can be easily implemented in an automatic program.

As an example, we consider this simple dynamic model, whose LFT formulation will be derived

$$\begin{cases} \dot{x}_1 = d_1 x_1 + \frac{x_2}{1+d_2} \\ \dot{x}_2 = u_1 \end{cases} \tag{6.22}$$

where d are the uncertain parameters.

The steps are the following:

1. Parameters that have to be estimated are put in $\delta = \begin{bmatrix} \delta_1 & \dots & \delta_q \end{bmatrix}$.

   Example: d_1 and d_2 are put into δ: $\delta = \begin{bmatrix} d_1 & d_2 \end{bmatrix}$.

2. Each δ has to multiply another variable. Otherwise, equations are modified to get the result.

   Example: $\dfrac{x_2}{1+d_2}$ becomes $\dfrac{x_2^2}{x_2 + x_2 d_2}$

3. Each x, ζ or w linearly dependent on a single δ (even at a denominator) is put in the z slot corresponding to the considered δ. Each $z\delta$ gets replaced by w.

   Example: we impose
   $z_1 = x_1 \quad z_2 = x_2$
   then
   $w_1 = z_1\delta_1 \quad w_2 = z_2\delta_2$

4. Each nonlinear part dependent on one or multiple x, u or w is put in a ζ, and each x, u or w needed in these parts is put in a ω.

   Example: since we have $\frac{x_2^2}{x_2 + x_2 d_2}$, we define
   $\omega_1 = x_2 \quad \omega_2 = w_2 = z_2\delta_2$
   then
   $\Theta(\omega) = \frac{\omega_1^2}{\omega_1 + \omega_2}$

5. The previous two steps are repeated until necessary.

If multiple ζ are present in a single equation, is useful to group them all in a single ζ, to simplify the system.

Here is reported the final LFT formulation corresponding to the example model (6.22)

$$z = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \qquad w = \begin{bmatrix} z_1\delta_1 & z_2\delta_2 \end{bmatrix} \tag{6.23}$$

$$\omega = \begin{bmatrix} x_2 & w_2 \end{bmatrix} \qquad \zeta = \Theta(\omega) = \frac{\omega_1^2}{\omega_1 + \omega_2} \tag{6.24}$$

$$\dot{x} = \begin{bmatrix} d_1 x_1 + \zeta_1 & u_1 \end{bmatrix} \tag{6.25}$$

The resulting equations then have to be translated into the matrix formulation (6.21), reported again for simplicity purposes, by properly filling the constant matrices

$$\begin{cases} \dot{x}(t) = Ax(t) + B_1 w(t) + B_2 \zeta(t) + B_3 u(t) \\ z(t) = C_1 x(t) + D_{11} w(t) + D_{12} \zeta(t) + D_{13} u(t) \\ \omega(t) = C_2 x(t) + D_{21} w(t) + D_{22} \zeta(t) + D_{23} u(t) \\ y(t) = C_3 x(t) + D_{31} w(t) + D_{32} \zeta(t) + D_{33} u(t) \\ w(t) = \Delta z(t) = diag\{\delta_1 I_{r_1}, ..., \delta_q I_{r_q}\} z(t) \\ \zeta(t) = \Theta(\omega(t)) \end{cases} \tag{6.26}$$

6.2.4 Using the LFT form as estimator

Given a system in the LFT formulation, the Hessian and gradient needed by a minimization method can be computed through a recently-developed two-pass algorithm [6], already applied to multiple problems [7] [8].

As already explained, Hessian and gradient of the parameter identification cost

function can be written as

$$H(J) = \frac{1}{N}\sum_{k=1}^{N} E^T(k,\delta)E(k,\delta) \tag{6.27}$$

$$\nabla J = \frac{1}{N}\sum_{k=1}^{N} E^T(k,\delta)e(k,\delta) \tag{6.28}$$

Using the definitions of prediction error and its Jacobian

$$e(k,\delta) = y(k) - \hat{y}(k,\delta) \tag{6.29}$$

$$E(k,\delta) = \frac{\partial e}{\partial \delta} = -\frac{\partial \hat{y}(k,\delta)}{\partial \delta} \tag{6.30}$$

we can replace them in (6.27) (6.28), obtaining

$$H(J) = \frac{1}{N}\sum_{k=1}^{N} \left(\frac{\partial y(k,\delta)}{\partial \delta}\right)^T \left(\frac{\partial \hat{y}(k,\delta)}{\partial \delta}\right) \tag{6.31}$$

$$\nabla J = \frac{1}{N}\sum_{k=1}^{N} \left(\frac{\partial y(k,\delta)}{\partial \delta}\right)^T \left[\hat{y}(k,\delta) - y(k)\right] \tag{6.32}$$

Both equations depend only on $y(k,\delta)$ (input measurement values), $\hat{y}(k)$ (predicted output) and $\frac{\partial \hat{y}(k,\delta)}{\partial \delta}$ (its Jacobian, or sensitivity): we can compute the last two using two different LFT formulations.

First stage

A new system can be built from the LFT model as follows

$$\begin{cases} M\dot{\hat{x}} = f(\hat{x},u) \\ y = g(\hat{x},u) \end{cases} \tag{6.33}$$

where

$$\hat{x} = [x^T \; z^T \; \omega^T]^T \tag{6.34}$$

$$M = \begin{bmatrix} I_n & 0_{n\times n_z} & 0_{n\times n_\omega} \\ 0_{n_z\times n} & 0_{n_z\times n_z} & 0_{n_z\times n_\omega} \\ 0_{n_\omega\times n} & 0_{n_\omega\times n_z} & 0_{n_\omega\times n_\omega} \end{bmatrix} \tag{6.35}$$

$$f(\hat{x},u) = \begin{bmatrix} Ax + B_1 w + B_2\zeta + B_3 u \\ C_1 x + (D_{11}\Delta - I_{n_z})z + D_{12}\zeta + D_{13}u \\ C_2 x + D_{21}w + D_{22}\zeta - \omega + D_{23}u \end{bmatrix} \tag{6.36}$$

$$g(\hat{x},u) = C_3 x + D_{31}w + D_{32}\zeta + D_{33}u \tag{6.37}$$

where n is the size of vector x, n_ω the size of ω, n_z the size of z.

This represents an index-1, semi-explicit Differential Algebraic Equations (DAE) system, that can be solved through a numerical integration method, obtaining $y = \hat{y}$.

Mass matrix M is singular, therefore the existence of a solution depends on providing a consistent initial state vector $\hat{\dot{x}}(0)$ [7]: in this way the first n equations are solvable and can provide a result.

Second stage

A second system can be built as follows

$$\begin{cases} \dot{x}_i' = Ax_i' + B_1 w_i' + B_2 \zeta_i + B_1 \Delta_{\delta_i} z \\ z_i' = C_1 x_i' + D_{11} w_i' + D_{12} \zeta_i' + D_{11} \Delta_{\delta_i} z \\ \omega_i' = C_2 x_i' + D_{21} w_i' + D_{22} \zeta_i' + D_{21} \Delta_{\delta_i} z \\ y_i' = C_3 z_i' + D_{32} w_i' + D_{32} \zeta_i + D_{31} \Delta_{\delta_i} z \\ w_i' = \Delta z_i' \\ \zeta_i' = \Theta_\omega(\omega) \omega_i' \end{cases} \tag{6.38}$$

where subscript i and superscript $'$ denote a new set of vectors. The nonlinear function vector has been replaced by its derivative with respect to each ω

$$\Theta_\omega(\omega) = \frac{\partial \Theta(\omega)}{\partial \omega} \tag{6.39}$$

while matrix Δ is equivalent to the one of the original system, and

$$\Delta_{\delta_i} = \frac{\partial \Delta}{\partial \delta_i} = diag(0, ..., I_{r_i}, ..., 0) \tag{6.40}$$

The last two equations are solved, obtaining $w(z)$ and $\zeta(\omega)$: the results are used to solve the 2nd and 3rd equation, obtaining ω and z; finally, w, ζ, ω are replaced in the 1st and 4th equation, obtaining a linear, time-varying system

$$\begin{cases} \dot{x}_i' = \hat{A}(\omega) x_i' + \hat{B}(\omega) \Delta_{\delta_i} z \\ y_i' = \hat{C}(\omega) x_i' + \hat{D}(\omega) \Delta_{\delta_i} z \end{cases} \tag{6.41}$$

where

$$\hat{A}(\omega) = A + \begin{bmatrix} B_1\Delta & B_2\Theta_\omega(\omega) \end{bmatrix} W(\omega) \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{6.42}$$

$$\hat{B}(\omega) = B_1 + \begin{bmatrix} B_1\Delta & B_2\Theta_\omega(\omega) \end{bmatrix} W(\omega) \begin{bmatrix} D_{11} \\ D_{21} \end{bmatrix} \tag{6.43}$$

$$\hat{C}(\omega) = C_3 + \begin{bmatrix} D_{31}\Delta & D_{32}\Theta_\omega(\omega) \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{6.44}$$

$$\hat{D}(\omega) = D_{31} + \begin{bmatrix} D_{31}\Delta & D_{32}\Theta_\omega(\omega) \end{bmatrix} \begin{bmatrix} D_{11} \\ D_{21} \end{bmatrix} \tag{6.45}$$

$$W(\omega) = \begin{bmatrix} I_{n_z} - D_{11}\Delta & -D_{12}\Theta_\omega(\omega) \\ -D_{21}\Delta & I_{n_\omega} - D_{22}\Theta_\omega(\omega) \end{bmatrix}^{-1} \tag{6.46}$$

Solving system (6.41) is equivalent to computing $y' = \frac{\partial \hat{y}(k,\delta)}{\partial \delta_i}$.

Since the second stage requires the value of $\omega(t)$ computed in the first stage, the two systems must be run in cascade, as in Figure 6.2, that shows the complete procedure.

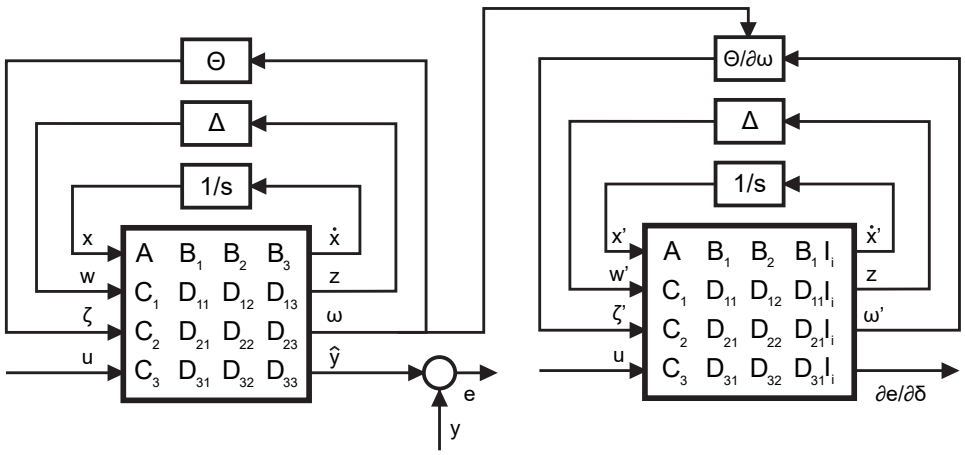


Figure 6.2: Two-stage LFT-based algorithm for parameter identification

There are a couple of problems that must be considered when implementing the algorithm. First of all, time instants selected by the solver to integrate respectively the first and second stage are usually different, making impossible the usage of ω by the second stage. A solution consists in rewriting (6.41) as

$$\dot{x}_i' = \Gamma(\omega) \begin{bmatrix} x_i' \\ \Delta_{\delta_i} z \end{bmatrix} \tag{6.47}$$

$$\Gamma(\omega) = \begin{bmatrix} \hat{A}(\omega) & \hat{B}(\omega) \end{bmatrix} \tag{6.48}$$

Thus isolating ω into Γ. The latter is computed with the $\omega(t)$ used by the first stage, saving the values and their times. When the second stage needs Γ, it is calculated by interpolating available data with the right time instant.

Another problem is related to the identificability of a parameter: a $\hat{\delta}$ is locally identificable if the Hessian $\hat{H}$ is positive definite in $\hat{\delta}$, i.e. if its rank is equal to $dim(\delta) = q$. Otherwise the estimated parameters are not unique. In this case, the Hessian is decomposed in two subspaces, one of which identificable

$$\hat{H} = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} \tag{6.49}$$

where U_1 is the subspace of δ that can be identified from the measures, U_2 the non-identifiable subspace, Σ_1 and Σ_2 are arbitrary matrices chosen to split singular values from the others, and V the rest of the system.

The tool can now be coupled with an optimization method to estimate the uncertainties of the system. It is important to emphatize the fact that while this approach strongly reduces the computational cost linked to the Hessian derivation, the optimization time is still strongly dependent on the numerical algorithm, that must be chosen accordingly to the needed application and eventually limited in its execution time, if bandwidth is a constraint.

6.3 The LFT Toolbox

6.3.1 Description

The previously described two-stage estimator is available in a ready-to-use Matlab implementation [6], still maintained as 2017. While the code is intended for processing Eclipse-generated problems, it provides two main functions for a generic usage:

- $[output] = lftSolver(lftfun, input, initialConditions, options)$

  solves a system in LFT formulation, given its LFT matrices, nonlinear functions, derivatives of the nonlinear functions with respect to each variable ω, inputs, initial conditions, uncertain parameter values, timespan.

- $[delta] = lftOptDelta(lftfun, Input, InitialConditions, solverOptions, y, optimOptions)$

  run the two-stage estimator on the given LFT system, performing the minimization of the error between the system output and a provided set of experimental data y.

The integration numerical method is provided by *ode15s*, a Matlab-provided 1st-5th order method with a medium resolution [3], while the minimization numerical

method is provided by *fminunc*, that by default uses a trusted-region algorithm. Both steps are tunable through a set of options.

An interesting feature consists in the possibility of weighting the error cost relative to each available measurement; in case of quadratic cost functions (like our objective, that is still equation (6.2)), this operation is usually performed through a weighting matrix ($x^T Q x$), but the toolbox uses this alternative method

$$e(k,\delta) = \left[\frac{y_1(k) - \hat{y}_1(k,\delta)}{W_1} \quad \dots \quad \frac{y_n(k) - \hat{y}_n(k,\delta)}{W_q} \right] \tag{6.50}$$

$$E(k,\delta) = \left[-\frac{\partial \hat{y}_1(k)/\partial\delta}{W_1} \quad \dots \quad -\frac{\partial \hat{y}_q(k)/\partial\delta}{W_q} \right] \tag{6.51}$$

where W is a vector, with length q equal to the one of the available measurements vector y. A larger value in a certain index corresponds to a lower weight of the error between the corresponding measurement and the system output, and vice-versa. This options will be useful when dealing with data from different sensors.

6.3.2 Improvements

The available toolbox already provides the entire estimation procedure, but a couple of improvements can be implemented with the aim of avoiding useless calculations and easing the optimization, before proceeding to a possible application.

Beside the listed features, also the derivation of the Jacobian of the nonlinear function vector Θ can be automatized through the symbolic toolbox. This part, however, is not reported since it is trivial.

Parameter normalization

It is important that all the members of the optimizing parameter vector δ stay as much as possible in the range ± 1 during the estimation, from initial value to optimal value [7]. While a reason is not provided by available literature, it is probably to be found in the necessity of inverting the Hessian matrix, an operation performed by the numerical methods, as very different optimizing parameter values can lead to high condition numbers and so to to an ill-formed problem. Normalized parameters $\bar{\delta} \in [-1; 1]$ can be introduced through this definition

$$\delta_i = \frac{\delta_{i,max} + \delta_{i,min}}{2} + \frac{\delta_{i,max} - \delta_{i,min}}{2} \bar{\delta}_i \tag{6.52}$$

where $\delta_{i,max}$ and $\delta_{i,min}$ are the boundaries of a reasonable range in which the uncertainties will be searched. This additional step can be performed manually, obtaining more complex LFT formulations [8] [10]; in this work an automatic procedure

is derived. By introducing the previous definition in the w equation of the LFT formulation (6.21), we obtain

$$w = diag(\delta)z = diag\left(\frac{\delta_{i,max} + \delta_{i,min}}{2} + \frac{\delta_{i,max} - \delta_{i,min}}{2}\bar{\delta}_i\right)z = \\ = Ez + diag(\bar{\delta})Fz \quad (6.53)$$

where we have defined, for simplicity reasons

$$E = diag\left(\frac{\delta_{i,max} + \delta_{i,min}}{2}, ...\right) \quad (6.54)$$

$$F = diag\left(\frac{\delta_{i,max} - \delta_{i,min}}{2}, ...\right) \quad (6.55)$$

we define the transformed w and z as

$$\tilde{w} = diag(\bar{\delta})Fz = diag(\bar{\delta})\tilde{z} \quad (6.56)$$

$$\tilde{z} = Fz \quad (6.57)$$

and replace $\tilde{w}$ in the w equation

$$w = Ez + \tilde{w} \quad (6.58)$$

w is replaced in the other equations

$$\begin{cases} \dot{x} = Ax + B_1Ez + B_1\tilde{w} + B_2\zeta + B_3u \\ z = C_1x + D_{11}Ez + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u \\ \omega = C_2x + D_{21}Ez + D_{21}\tilde{w} + D_{22}\zeta + D_{23}u \\ y = C_3x + D_{31}Ez + D_{31}\tilde{w} + D_{32}\zeta + D_{33}u \end{cases} \quad (6.59)$$

z can be solved from the 2nd equation

$$z = (I - D_{11}E)^{-1}[C_1x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] = \\ = G[C_1x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] \quad (6.60)$$

where we have defined, for simplicity reasons

$$G = (I - D_{11}E)^{-1} \quad (6.61)$$

z is then replaced in the other equations

$$\begin{cases} \dot{x} = Ax + B_1EG[C_1x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] + B_1\tilde{w} + B_2\zeta + B_3u \\ \omega = C_2x + D_{21}EG[C_1x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] + D_{21}\tilde{w} + D_{22}\zeta + D_{23}u \\ y = C_3x + D_{31}EG[C_1x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] + D_{31}\tilde{w} + D_{32}\zeta + D_{33}u \end{cases} \quad (6.62)$$

z is also replaced in $\tilde{z}$

$$\tilde{z} = FG[C_1 x + D_{11}\tilde{w} + D_{12}\zeta + D_{13}u] \tag{6.63}$$

By merging (6.56), (6.62) and (6.63) and rearranging the terms, we obtain

$$\begin{bmatrix} \dot{x} \\ \tilde{z} \\ \omega \\ y \end{bmatrix} = \begin{bmatrix} A + B_1 EGC_1 & B_1 + B_1 EGD_{11} & B_2 + B_1 EGD_{12} & B_3 + B_1 EGD_{13} \\ FGC_1 & FGD_{11} & FGD_{12} & FGD_{13} \\ C_2 + D_{21} EGC_1 & D_{21} + D_{21} EGD_{11} & D_{22} + D_{21} EGD_{12} & D_{23} + D_{21} EGD_{13} \\ C_3 + D_{31} EGC_1 & D_{31} + D_{31} EGD_{11} & D_{32} + D_{31} EGD_{12} & D_{33} + D_{31} EGD_{13} \end{bmatrix} \begin{bmatrix} x \\ \tilde{w} \\ \zeta \\ u \end{bmatrix} \tag{6.64}$$

This system allows to obtain the matrices of to a new, normalized LFT formulation, i.e. $\tilde{A} = A + B_1 EGC_1$.

The transformation can be applied to the nominal matrices before using the toolbox, avoiding a manual calculation.

Constrained search

The considered numerical minimization methods work well in most of the cases, but repeated testings have revealed that the estimator is not very robust against measurements with lot of noise, or measurements relative to a phenomenon that does not resemble enough the considered model, or in case of multiple uncertain parameters, that require a high number of iterations to converge, or in case of wrong initial conditions. If the solver does not properly detect the search direction, it could decide to increase the step size α or the optimizing parameters indefinitely, probably incurring in a singularity and exiting without giving results.

A solution to increase the robustness of the estimator consists in switching from an unconstrained optimization procedure to a hard-constrained optimization procedure, that does not allow the parameters to exit a certain range, set by one or more inequality constraints. This is obtained in literature by augmenting the cost function with a equality constraint, through a coefficient μ (adjoint vector) [39]

$$J_A = J_{orig} + \mu_{eff} c_{eff} \tag{6.65}$$

where c_{eff} is a equality constraint in the form $c(\delta) = 0$, that is active when the inequality constraint limits are reached. This constraint can be $-1 \leq \delta \leq 1$, forcing the unknown parameters to assume a value inside the normalized range.

To introduce the feature, it is enough to replace the standard Matlab nonlinear optimization function *fminunc* with *fmincon* (trust region reflective algorithm), providing needed parameters.

6.4 Estimation with position measurements

It is now possible to build a sideslip estimator based on the explained concepts. The tool explained in this section is similar to the one already implemented in previous works [8] [10], although built on a different model, and is analyzed as a starting point for possible improvements.

6.4.1 Formulation

We recall the single track model (equations (3.20)), written in this canonical form

$$\begin{cases} \dot{v} = \frac{1}{m}[C_f(\frac{-v}{u} - a\frac{r}{u} + G\delta) + C_r(\frac{-v}{u} + b\frac{r}{u})] - ur \\ \dot{r} = \frac{1}{I_z}(aC_f(\frac{-v}{u} - a\frac{r}{u} + G\delta) - bC_r(\frac{-v}{u} + b\frac{r}{u})) \\ \dot{\psi} = r \\ \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \end{cases} \tag{6.66}$$

A beta estimation can be performed by simply simulating the model in real-time, obtaining the value through its definition

$$\beta = \arctan\left(\frac{v}{u}\right) \tag{6.67}$$

We now need to decide what is fed into the model during the simulation and what instead needs to be identified. Choices relative to the input and state vector are similar to the one performed for the EKF-based estimator in Chapter 4:

$$x = \begin{bmatrix} v & r & \psi & x & y \end{bmatrix}^T \tag{6.68}$$

$$u = \begin{bmatrix} u & \delta \end{bmatrix}^T \tag{6.69}$$

C_f and C_r are the critical uncertain parameters, due to their dependency on the road and vehicle specific conditions; a first tool can be built to identify only the stiffnesses, that are inserted in the unknown parameters vector δ

$$\delta = \begin{bmatrix} C_f & C_r \end{bmatrix}^T \tag{6.70}$$

The other model parameters, m, I_z, a and b, must be estimated separately and treated as constants. This will represent a problem, since any uncertainty will lead to an estimation error.

The last operation consists in choosing what kind of measurements the identification will be based on. As first a try, we can use the vehicle absolute position measurements, as they constitute an absolute reference that is always useful to

mitigate drift. The comparison measurements are linked to the state variables through the output vector y

$$y = \begin{bmatrix} x & y \end{bmatrix}^T \tag{6.71}$$

The LFT formulation can now be derived from the complete system, by using the procedure described in Section 6.2.3. Multiple formulations are possible and valid. A general rule consists in choosing the option with the less number of nonlinear functions $\Theta(\omega)$. The one used in this work is reported below.

$$x = \begin{bmatrix} v & r & \psi & x & y \end{bmatrix}^T \tag{6.72}$$

$$y = \begin{bmatrix} x_4 & x_5 \end{bmatrix}^T \tag{6.73}$$

$$\delta = \begin{bmatrix} C_f & C_r \end{bmatrix}^T \tag{6.74}$$

$$w = \begin{bmatrix} z_1\delta_1 & z_2\delta_2 \end{bmatrix}^T \tag{6.75}$$

$$\omega = \begin{bmatrix} x_1 & x_2 & x_3 & u_1 & u_2 & w_1 & w_2 \end{bmatrix}^T \tag{6.76}$$

$$z = \begin{bmatrix} \zeta_1 & \zeta_2 \end{bmatrix}^T \tag{6.77}$$

$$\zeta = \begin{bmatrix} \frac{-\omega_1}{\omega_4} - a\frac{\omega_2}{\omega_4} + G\omega_5 \\ \frac{-\omega_1}{\omega_4} + b\frac{\omega_2}{\omega_4} \\ \omega_4 \cos\omega_3 - \omega_1 \sin\omega_3 \\ \omega_4 \sin\omega_3 + \omega_1 \cos\omega_3 \\ \frac{1}{m}[\omega_6 + \omega_7] - \omega_4\omega_2 \end{bmatrix} \qquad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} \zeta_5 \\ \frac{1}{I_z}[aw_1 - bw_2] \\ x_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix} \tag{6.78}$$

From these equations it is possible to extract all the coefficients of the matrices present in system (6.21), and setup an identification procedure through the toolbox. Of course the real formulation used in the estimator is more complex, as it includes the automatic parameter normalization procedure, and is not reported for simplicity reasons.

6.4.2 Simulation results

The efficacy of this approach can now be verified by using available data, starting from the one obtained from the Dymola-Simulink test environment described in Appendix A. A few additional steps are needed, with respect to the feedback linearization-based estimator, before the estimator is ready to use.

To begin with, all the various static parameters need to be set or identified:

- Geometrical properties a and b can be read from the vehicle properties;
- Steering gain G and inertial properties m and I_z, referred to the center of mass, can be estimated through a grey-box procedure; in this case, Dymola provides a dedicated block.

An important aspect is the definition of a reasonable range in which the cornering stiffnesses will be identified, that will also be used to normalize the parameters, as explained in the dedicated section. This operating zone can be deducted from multiple tests or from vehicle properties. In this case, the estimated values reported in Appendix A have led to the choice of the ranges reported in the table below.

Front stiffness range	C_f	$10000 - 200000$	N/rad
Rear stiffness range	C_r	$10000 - 200000$	N/rad

Another required task before running the tool consists in filling the initial state vector of the LFT model: this operation is particularly important as it influences the convergence of the solution (in literature this is denoted as a major limitation of least squares-based estimators [17]). Values can be taken from sensors or previous simulations.

Roundabout trajectory

A first estimation is performed against data from the Dymola simulation of a vehicle running at $80\ km/h$ in the roundabout trajectory (Section A.3), with the steering wheel oriented at $45°$ for 30 seconds. Samples are limited in quantity (200 samples), not to result in an excessive computation time, and ideal position measurements are fed into the identification procedure, that is run once. After identification, the LFT model is simulated with the identified parameters, over the same timespan, obtaining beta.

The results, shown in Figure 6.3, already constitute a clear indicator of the main limitation of this approach: the value is characterized by a constant bias error, dependent on the choice of the static parameters m and I_z, that must be manually tuned to solve the problem.

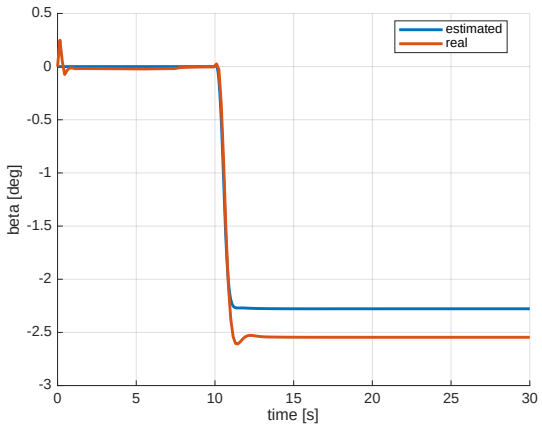
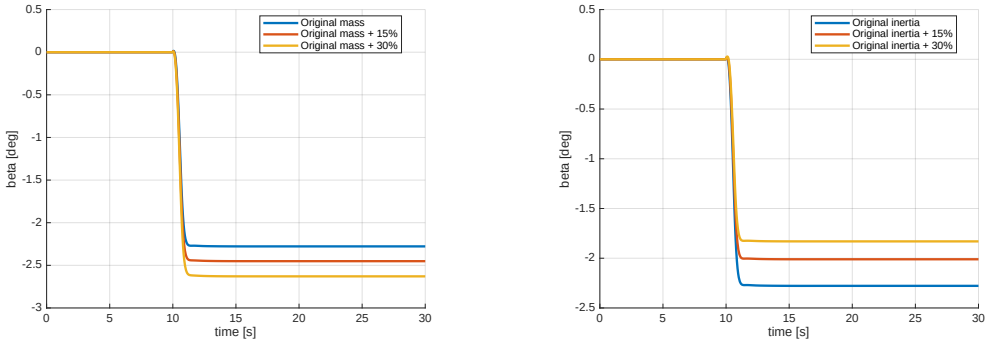


Figure 6.3: Results of the untuned LFT estimator against the roundabout trajectory

Parameter sensitivity

The dependency of the estimation from the static parameters can be analyzed in a more detailed way: the simulation can be repeated multiple times, varying mass and inertia values. It is possible to discover that a 30% increase of m causes a 20% variation of the estimated slip angle. An increase of I_z causes the opposite effect, as shown in Figure 6.4.



(a) Mass **(b)** Inertia

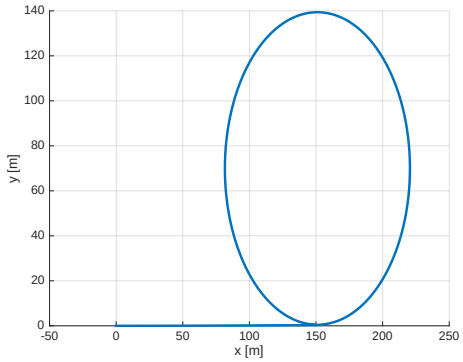
Figure 6.4: LFT estimator sensitivity to parameters variation

If slip angle measurements are available from a more precise method (for instance, feedback linearization) an option consists in setting m and I_z through a trial-and-error procedure, until the estimation error drops below a certain value. In this way, a working estimator can be obtained, however the resulting tool cannot take into account eventual variations of the parameters over time, a situation that can happen due to additional loading or solicitations. This constitutes a critical issue that prevents a real-world application of the approach.

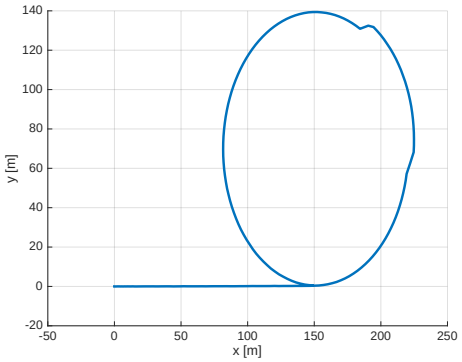
GNSS robustness

The estimation is characterized by a low sensitivity to measurements errors: their presence minimally influences the identification procedure, since it is based on the least squares minimization of an error function, and sudden measurement shifts are not reflected in the estimated beta value, since the latter is obtained from the simulation of a "perfect" model.

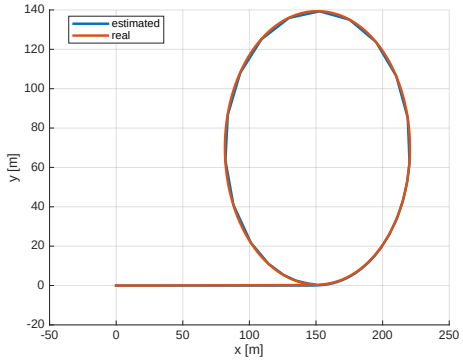
This behavior can be shown by introducing the previously-built GNSS error model into the position measurements, as shown in Figure 6.5b: the model state variables x and y, represented in Figure 6.5c, are able to track perfectly the real position, even if subjected to relevant shifts, resulting in a correct estimation.



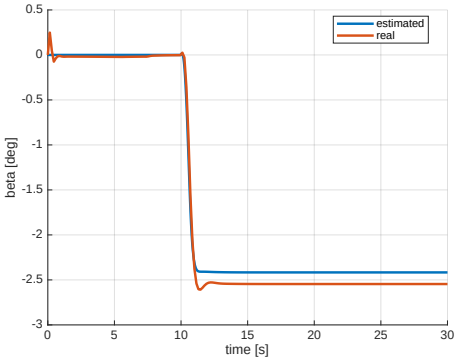
(a) Real vehicle position



(b) Position measurements



(c) Estimated position



(d) Estimated slip angle

Figure 6.5: Results of the LFT estimator against the GNSS error model

Slalom trajectory

A simulation of the slalom trajectory, shown in Figure 6.6, allows to demonstrate an already anticipated intrinsic limitation of this approach: a single estimation of the cornering stiffnesses is unable to account for the non-linearities that arise when the tyre solicitations exits outside a safe range. This limitation can only be addressed with an improvement of the system, like a better identification procedure or the introduction of a full tyre model, or by a repeated identification of the stiffnesses.

Another demonstration of the phenomenon can be performed by measuring the steady-state absolute error of an estimation against the roundabout path, for increasing longitudinal velocities (corresponding to increasing solicitations), as shown in Figure 6.7: the error rises almost linearly with the velocity, and so with the stress.

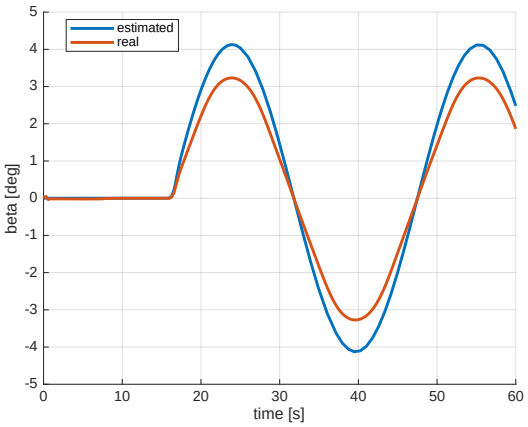


Figure 6.6: Results of the untuned LFT estimator against the slalom trajectory

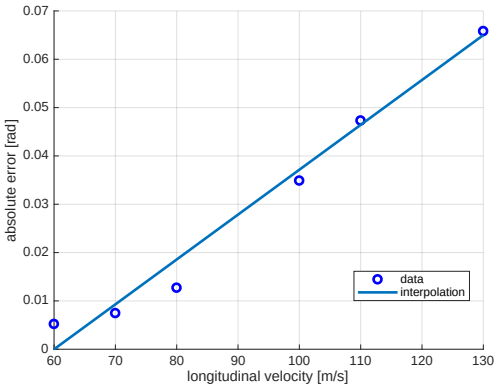


Figure 6.7: Absolute error of the LFT estimator for increasing velocities

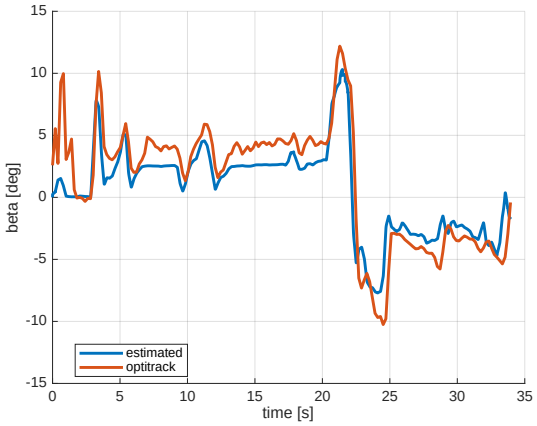
6.4.3 Experimental results

The discussed concepts were validated by running the estimator against measurements from sensors placed on the unmanned ground vehicle described in Appendix B. The identification ranges of the cornering stiffnesses are reported in the table below.

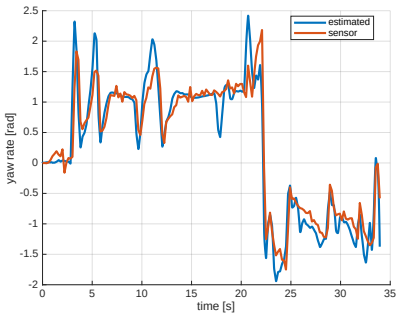
Front stiffness range	C_f	$10 - 60$	N/rad
Rear stiffness range	C_r	$10 - 60$	N/rad

Results provided by the tool are reported in Figure 6.8: the model is able to track the vehicle position and to obtain reasonable state values, in a way similar to the one of the EKF-based estimator. The main difference concerns the handling of

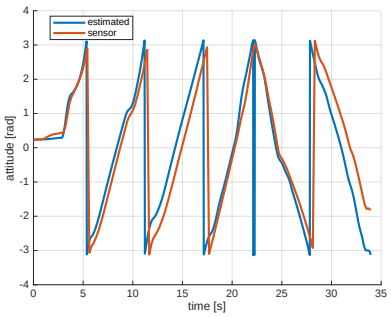
the uncertain parameters: while the EKF estimator vary these values during the simulation to compensate the errors, the LFT estimator identify a single set of parameters for each timespan, and then simulates the system.



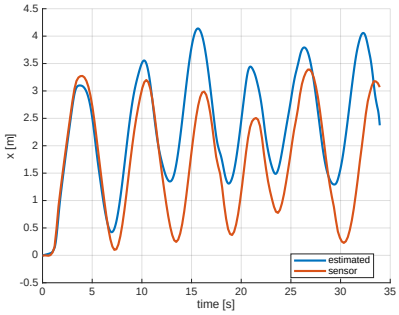
(a) Slip angle



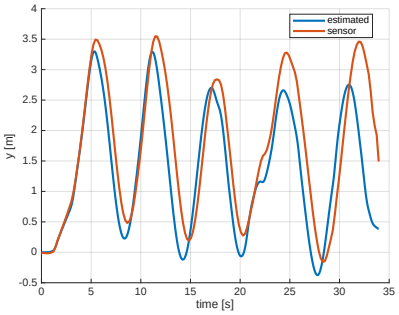
(b) Yaw rate



(c) Attitude



(d) Position (x axis)



(e) Position (y axis)

Figure 6.8: Results of the LFT estimator against data from a UGV

It is difficult to assert which of the two estimators is better, in terms of precision: the EKF is undoubtedly more sensitive to measurement errors, that are almost completely rejected by the LFT; on the other hand, the EKF seems to interpolate better the given measurements. This can be demostrated by computing the TIC statistic for each available measurement and output (equation (4.31)). Results

relative to the LFT-based estimator, reported in the table below, have an higher order of magnitude than the ones of the EKF-based estimator.

	LFT-based estimator	EKF-based estimator
TIC_x	0.084112	0.0011909
TIC_y	0.071208	0.00076763

6.4.4 Conclusions

The benefits of model-based estimators, that include a flexible bandwidth and a robustness against estimation errors, have lead to their consideration in most of the research activity dedicated to lateral vehicle dynamics.

The LFT approach for parameter identification undoubtedly constitutes an innovation with respect to the standard Extended Kalman Filter, in terms of computational cost and convergence. A weakness is constituted by the necessary choice of the initial state values, that influences the convergence and correctness of the estimation.

Nonetheless, limitations in the considered variant, that uses static vehicle mass and inertia, considers as uncertain only the cornering stiffnesses and exploits the position measurements in the identification procedure, critically influence the estimation precision. Static parameters have a large influence on the accuracy, and must be tuned to obtain a good result, by using as reference a more precise estimation system. Any variation in static parameters during the test introduce irremovable bias errors. If precision is not a relevant factor, like in case of some safety systems, the tool can be adopted, otherwise it must be improved.

Another factor that must be considered when choosing an estimation method, is that the numerical algorithms used in the optimization procedure do not have a fixed execution time, so they are not deterministic. This can constitute a serious limitation if the aim is implementing the algorithm in a hard real time system. The feedback linearization technique also uses numerical algorithms for integration, but these are generally lighter than numerical algorithms for optimization; moreover, there exists some deterministic integration methods, that can be used if a fixed execution time is required. This limitation involves only the identification step, and not the estimation, so it can be mitigated by increasing the period between identifications.

Chapter 7

Improved slip angle estimation through LFT

This chapter is dedicated to some of the possible improvements that can be performed on the LFT-based sideslip estimator: by considering additional measurements in the identification procedure or by introducing more uncertain parameters, precision and flexibility of the results can be sensibly influenced. A series of variants, that consider the presence of an on-board IMU sensor, are presented and tested against available data.

7.1 Motivations

The limitations of a slip angle observer based only on position measurements have been explained, but at the same time a full description of the LFT-based parameter identification procedure has been provided, allowing the development of alternative variants of the algorithm, that can mitigate the precision problem.

Multiple paths can be followed, since a model-based estimator, coupled with a parameter identification technique, is a very flexible tool. Some of the available options are listed below:

- The model complexity can be increased, as more parameters can be introduced to better describe the phenomenon. This operation can be performed through one of the available tyre models, as multiple EKF-based estimators already do.
- The number of system outputs can be increased, allowing an estimation based on measurements from different sensors, that can have very different characteristics in terms of precision and bandwidth.
- More parameters can be estimated online, for instance the mass or the inertia of the vehicle, that constitute a major weakness of the base LFT estimator, as already discussed.

The last two possibilities are addresses in this chapter, in particular:

- An estimator is built to consider in the identification process also the values provided by an IMU sensor, placed at the center of mass of the vehicle. IMU values, as already introduced, are very useful when dealing with forces and accelerations, but cause progressive errors when the quantities are integrated.
- An estimator is built to identify, beside the cornering stiffnesses, also the mass and inertia.

This are just two of the possible variants of the algorithm, that can be arbitrarily increased in complexity and uncertain parameters, but whose convergence depends on the ability of available data of correctly identify all the parameters. As this work has the scope of developing a supporting tool for a control algorithm that will be applied to a small unmanned ground vehicle, the only considerable inputs are the one of a IMU and GNSS sensors placed at the center of gravity. This set of data is unfortunately too small to estimate the parameters of a full tyre model.

To implement the tool in an on-board processing unit, we also have to "discretize" the estimation, as it requires a set of samples over a certain time window. This time window must be defined, and influences the solution convergence. Problems linked to this variant are addressed in the last sections of the work.

7.2 Estimation with position and IMU measurements

7.2.1 Formulation

An improved estimator can be built by introducing in the identification process measurements from both a position and IMU sensor, placed at the center of gravity of the vehicle. The possible benefits include not only a more precise estimation of the stiffnesses, but also an increased robustness against GNSS sensor errors due to weak or absent signals, a very frequent situation, that is also the limiting factor of the feedback linearization-based estimator.

A 6-DOF IMU sensor provides measurements of acceleration and angular rate in 3 directions, but only the yaw rate and lateral acceleration are matter of interest for the lateral vehicle dynamics. In particular, the lateral acceleration directly provides a lateral velocity estimation through

$$a = \dot{v} + ur \tag{7.1}$$

where a is the lateral acceleration, v the lateral velocity, r the yaw rate, u the longitudinal velocity and $ur = u^2/R$ is the quantity associated to the centripetal force caused by a curved trajectory with radius R. This definition alone is usually hard to exploit, since it requires an integration, that in case of data from an IMU

sensor causes a progressive drift error. Instead, we can link a to the quantities of the considered model, avoiding a direct integration: by replacing $\dot{v}$ with the definition provided by the single track model (equations (6.66)), the formula results

$$a = \frac{1}{m}\left[C_f\left(\frac{-v}{u} - a\frac{r}{u} + G\delta\right) + C_r\left(\frac{-v}{u} + b\frac{r}{u}\right)\right] \tag{7.2}$$

We can finally add both the yaw rate (state variable r) and the lateral acceleration to the output vector y, to consider them in the identification process

$$y = \begin{bmatrix} x \\ y \\ r \\ \frac{1}{m}\left[C_f\left(\frac{-v}{u} - a\frac{r}{u} + G\delta\right) + C_r\left(\frac{-v}{u} + b\frac{r}{u}\right)\right] \end{bmatrix} \tag{7.3}$$

The LFT formulation can then be modified accordingly. The one used in the experimental tests is reported below.

$$x = \begin{bmatrix} v & r & \psi & x & y \end{bmatrix}^T \tag{7.4}$$

$$y = \begin{bmatrix} x_4 & x_5 & x_2 & \zeta_6 \end{bmatrix}^T \tag{7.5}$$

$$\delta = \begin{bmatrix} C_f & C_r \end{bmatrix}^T \tag{7.6}$$

$$w = \begin{bmatrix} z_1\delta_1 & z_2\delta_2 \end{bmatrix}^T \tag{7.7}$$

$$\omega = \begin{bmatrix} x_1 & x_2 & x_3 & u_1 & u_2 & w_1 & w_2 \end{bmatrix}^T \tag{7.8}$$

$$z = \begin{bmatrix} \zeta_1 & \zeta_2 \end{bmatrix}^T \tag{7.9}$$

$$\zeta = \begin{bmatrix} \frac{-\omega_1}{\omega_4} - a\frac{-\omega_2}{\omega_4} + G\omega_5 \\ \frac{-\omega_1}{\omega_4} + b\frac{\omega_2}{\omega_4} \\ \omega_4\cos\omega_3 - \omega_1\sin\omega_3 \\ \omega_4\sin\omega_3 + \omega_1\cos\omega_3 \\ \frac{1}{m}[\omega_6 + \omega_7] - \omega_4\omega_2 \\ \frac{1}{m}[\omega_6 + \omega_7] \end{bmatrix} \qquad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} \zeta_5 \\ \frac{1}{I_z}[aw_1 - bw_2] \\ x_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix} \tag{7.10}$$

7.2.2 Simulation results

Roundabout trajectory

A first estimation is performed by using the same data tested against the previous estimator (vehicle running at 80 km/h, steering wheel at 45°, roundabout trajectory, 30 s), providing ideal measurements to the identification procedure.

Results in Figure 7.1 show again the presence of a bias error in the estimation, mitigated with respect to the base variant of the algorithm: apparently the intrinsic uncertainties of the model parameters cannot be fully compensated with this method.

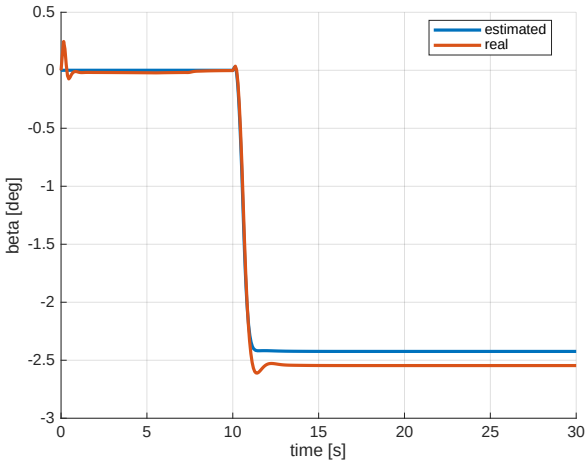


Figure 7.1: Results of the IMU-based estimator against the roundabout trajectory

Parameter sensitivity with respect to static parameters is also similar to the previous case: any change correspond to an irreversible bias error.

IMU role at high speeds

Nonetheless, an interesting phenomenon can be observed by progressively increasing the cruise velocity: the estimation becomes more and more precise, due to the fact that the lateral velocity detected by the accelerometer, that constitutes a direct link to the slip angle, progressively increases and becomes more and more relevant in the identification procedure. This is confirmed by plotting the absolute measurement error for increasing velocities, as in Figure 7.2. It seems like the static parameters influence only the role of the position measurements in the estimation, and not the IMU one.

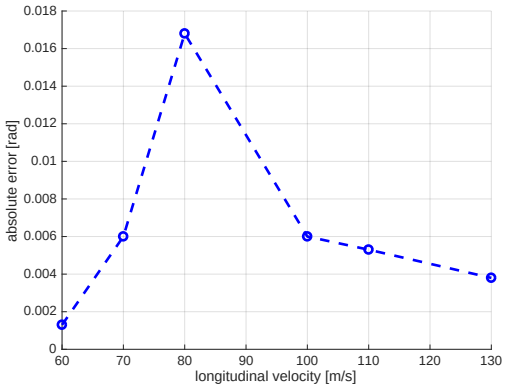


Figure 7.2: Absolute error of the IMU-based LFT estimator for increasing velocities

It is important to specify that the unknown parameters C_f and C_r now have a different physical meaning: while they are intended to represent the cornering stiffnesses, as defined in the Pacejka formula, now they are chosen to make the model reflect the lateral dynamics phenomenon, even if it is very different from the representation given by the equations. Anyway, this approach should work, because the IMU measurements provide a direct link to the sideslip angle, as represented in equation (7.1), and their contribution is predominant, with respect to position measurements, for increasing slip angles.

Error convergence can be enhanced by tuning the measurement weighting vector introduced by the toolbox (equations (6.51)-(6.50)), assigning a greater weight to the IMU measurements, in particular to the yaw rate, otherwise difficult to track adequately. A proper weighting vector could be

$$W = \begin{bmatrix} 10 & 10 & 10^{-3} & 10^{-1} \end{bmatrix} \tag{7.11}$$

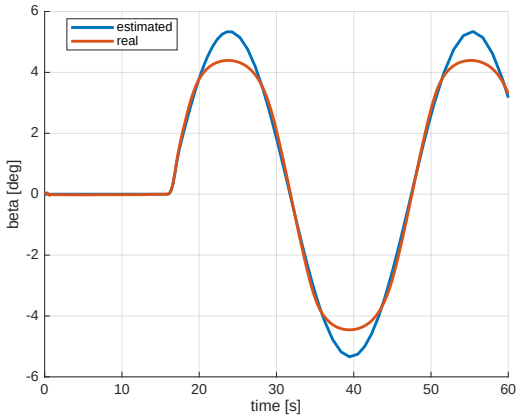
where each term is referred to the corresponding output defined in vector (7.3), reported again for simplicity purposes

$$y = \begin{bmatrix} x & y & r & a \end{bmatrix} \tag{7.12}$$

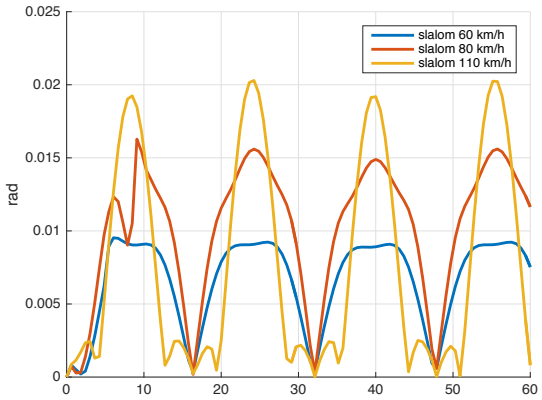
Unfortunately this tuning is not enough to eliminate the bias error at low longitudinal speeds, due to the fact that static parameters influence too much the model behavior in this operating range.

Slalom trajectory

The tracking of a slalom trajectory at high speeds is also performed better than the original tool, but another difficulty arises, as the passage between the linear zone and the nonlinear zone in the relationship between C and F_y (Figure 3.2) at high sideslip values cannot be reproduced by the model, and the saturation is not detected, as shown in Figure 7.3.



(a) Estimation



(b) Absolute error

Figure 7.3: Results of the IMU-based estimator against the slalom trajectory

The plot of the absolute estimation error for different velocities in Figure 7.3 summarizes the limitations of this approach: higher velocities correspond to a lower error, but the estimation saturates after a certain limit.

Estimation without GNSS signal

The behavior of a least squares-based estimator against weak GNSS signals has already been tested, and the considered variant is not different with regard to this aspect. The introduction of IMU measurements should result in an additional degree of robustness, that consists in the possibility of performing a limited estimation in case position measurements are completely unavailable.

By discarding the position measurements in the identification process, the estimation error results very low, as show in Figure 7.4, but a progressive drift error arises and is clearly visible.

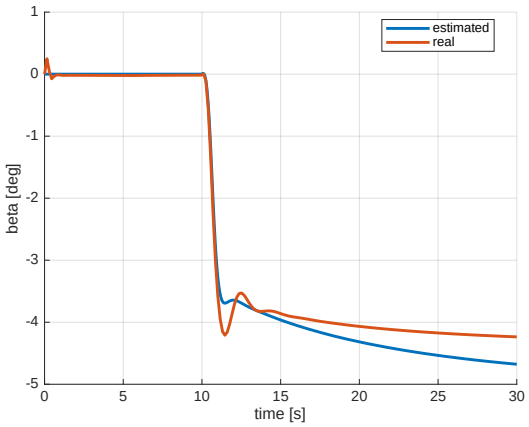


Figure 7.4: Results of the IMU-based LFT estimator in case of absent GNSS signal

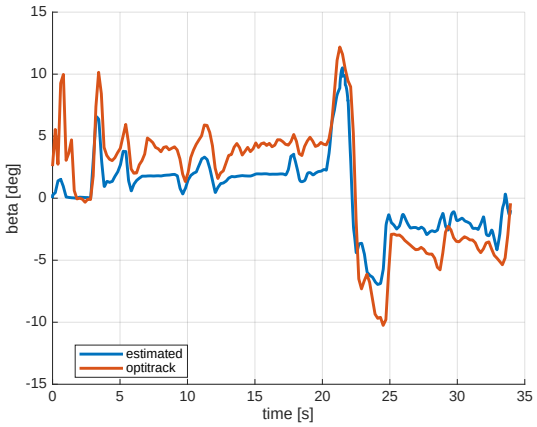
This behavior is typical of estimations based only on inertial sensors, that are based on integrations and lacks an absolute reference, and prevents their use as standalone tools for this kind of measurements.

7.2.3 Experimental results

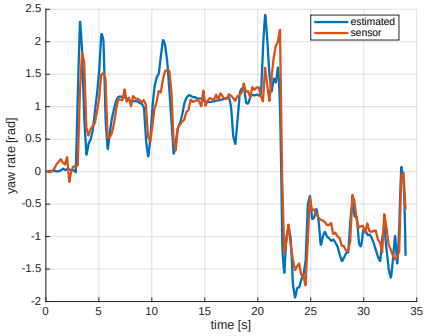
Tests can now be repeated against data from the Unmanned ground vehicle described in Appendix B, and in particular against the data set used throughout the work.

Results are shown in Figure 7.5: while the beta estimation is similar to the one of the previous case, the introduction of the lateral acceleration and yaw rate lead to a slightly better interpolation of the data set, with respect to the LFT estimator based exclusively on position measurements, as demostrated by the TIC statistics reported in the table below.

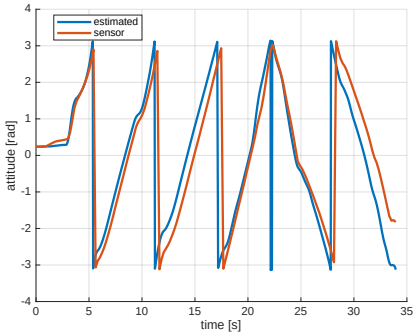
	position and IMU-based LFT estimator	position-based LFT estimator
TIC_x	0.082263	0.084112
TIC_y	0.06969	0.071208
TIC_ω	0.040071	-
TIC_{a_y}	0.096856	-



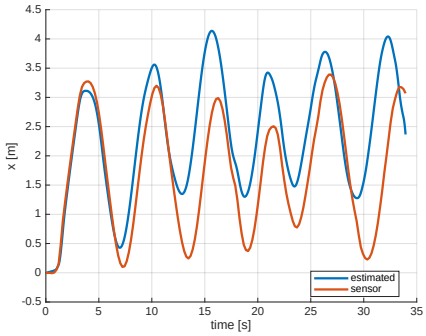
(a) Slip angle



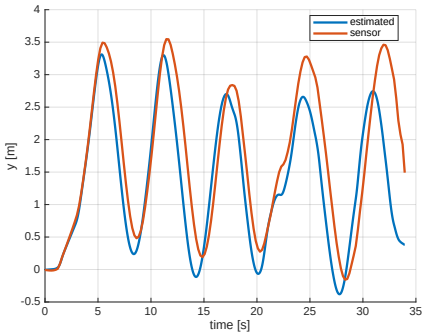
(b) Yaw rate



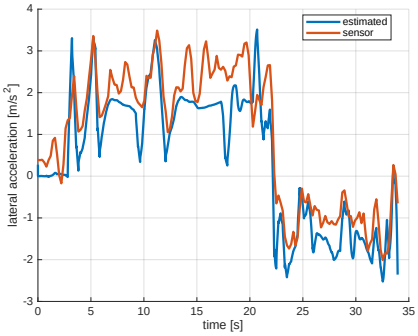
(c) Attitude



(d) Position (x axis)



(e) Position (y axis)



(f) Lateral acceleration

Figure 7.5: Results of the IMU-based LFT estimator against data from a UGV

7.2.4 Conclusions

The introduction of IMU measurements in the estimation procedure is not enough to compensate the uncertainties introduced by the static parameters of the single track model. Nonetheless, the tool can output reliable values at high speeds (in case of a repeated estimation), so it is although an improvement.

The robustness of the tool in case of absent or weak GNSS signal constitutes a significant improvement, but position measurements cannot be fully eliminated from the estimation, otherwise a drift error arises and makes the system useless.

7.3 Estimation with 4 uncertain parameters

7.3.1 Formulation

An attempt to overcome the main limitations of the state of the art LFT-based sideslip angle estimator, that are linked to its dependency on static parameters and their uncertainties, can be performed by introducing in the identification procedure also the mass and inertia of the vehicle, testing if the available measurements from a position and IMU sensor are enough to compute a proper estimation.

C_f, C_r, m and I_z now have to be identified together prior to the model simulation, and are inserted in the unknown parameters vector δ

$$\delta = \begin{bmatrix} C_f & C_r & m & I_z \end{bmatrix}^T \tag{7.13}$$

The LFT formulation can now be derived as usual, but some additional steps are required:

- The state equation corresponding to r, that includes the new uncertain parameters I_z at the denominator, has to be moved in vector ζ, as it is not linear anymore.
- A multiplying factor z must be chosen for the new parameters. Since they both appear at the denominator, the best choice would consist in using the inverse of their numerator. This causes an integration error with the current numerical method ($ode15s$) so an alternative consists in using a "1" as multiplying factor, imposed through an additional nonlinear function in ζ.
- The presence of uncertain parameters at a denominator causes the current implementation of the estimator to exit prematurely, since it detects a singularity at the initial point. A workaround consists in summing a small quantity to the variable corresponding to the uncertain parameters in the equations, avoiding the singularity.

The used formulas are reported below.

$$x = \begin{bmatrix} v & r & \psi & X & Y \end{bmatrix}^T \tag{7.14}$$

$$y = \begin{bmatrix} x_4 & x_5 & x_2 & \zeta_6 \end{bmatrix}^T \tag{7.15}$$

$$\delta = \begin{bmatrix} C_f & C_r & m & I_z \end{bmatrix}^T \tag{7.16}$$

$$w = \begin{bmatrix} z_1\delta_1 & z_2\delta_2 & z_3\delta_3 & z_4\delta_4 \end{bmatrix}^T \tag{7.17}$$

$$\omega = \begin{bmatrix} x_1 & x_2 & x_3 & u_1 & u_2 & w_1 & w_2 & w_3 & w_4 \end{bmatrix}^T \tag{7.18}$$

$$z = \begin{bmatrix} \zeta_1 & \zeta_2 & \zeta_7 & \zeta_7 \end{bmatrix}^T \tag{7.19}$$

$$\zeta = \begin{bmatrix} \frac{-\omega_1}{\omega_4} - a\frac{-\omega_2}{\omega_4} + G\omega_5 \\ \frac{-\omega_1}{\omega_4} + b\frac{\omega_2}{\omega_4} \\ \omega_4 \cos\omega_3 - \omega_1 \sin\omega_3 \\ \omega_4 \sin\omega_3 + \omega_1 \cos\omega_3 \\ \frac{1}{0.001+\omega_8}[\omega_6 + \omega_7] - \omega_4\omega_2 \\ \frac{1}{0.001+\omega_8}[\omega_6 + \omega_7] \\ 1 \\ \frac{1}{0.001+\omega_9}[a\omega_6 - b\omega_7] \end{bmatrix} \qquad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} \zeta_5 \\ \zeta_8 \\ x_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix} \tag{7.20}$$

The proposed identification procedure requires far more computational power than the other considered methods. Furthermore, the minimization algorithm is very critical with respect to the other cases, and a series of factors that before were marginal now assume a relevant role:

- The minimization can seldom be performed with success without the addition of hard constraints on the optimizing parameter values, described in Section 6.3.2.
- Initial guesses of the parameters must be sufficiently near their real value, otherwise the iteration results in a wrong estimation.
- Even in case of a constrained search, convergence depends on the specific combination of initial values, weighting matrix and tolerances of numerical algorithms, values that often need to be changed arbitrarily to unstuck the optimization algorithm.
- After finding a working combination, the tolerance can be decreased to lower the estimation error.

Nonetheless, the tool can be run after a proper tuning.

7.3.2 Simulation results

The additional uncertain parameters require the choice of proper ranges. Values relative to the simulation environment, in addition to the one already reported in Section 6.4.2, are reported in the table below: they are left big enough to allow the estimator to properly detect a search direction.

Mass range	m	$100 - 5000$	kg
Inertia range	I_z	$100 - 5000$	kgm

Roundabout trajectory

An estimation performed against the same data set used with the previous estimators (vehicle running at $80\ km/h$, steering wheel at $45°$, roundabout trajectory, $30\ s$) shows that the tool is indeed able to identify a set of parameters that, inserted in the vehicle dynamics model, can correctly estimate the slip angle at almost any velocity (before saturation occurs), as shown in Figure 7.6.

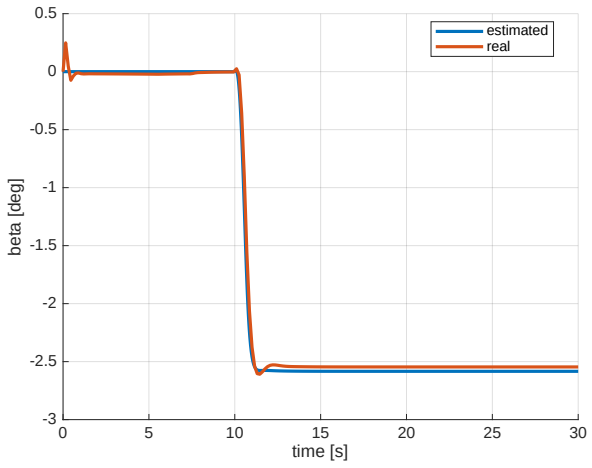


Figure 7.6: Results of the 4-parameters LFT estimator against the roundabout trajectory

The error decreases with the minimization procedure tolerance, that must be significantly less than the one used on other methods.

Like in the previous case, unknown parameters are identified not to respect their physical meaning (m is not chosen to represent the mass), but to make the model output reflect the measurements, even if it does not reflect well the vehicle behavior. This can be performed without particular problems for limited timespans, but the efficacy of the tool must be verified for each specific case.

The plot of the absolute error for increasing velocities, represented in Figure 7.7, apparently show that this last variant of the LFT estimator is able to exploit the advantages of both the IMU-based 2-parameters estimator and the GNSS-based 2-parameters estimator, by covering to the entire operating range with a low error.

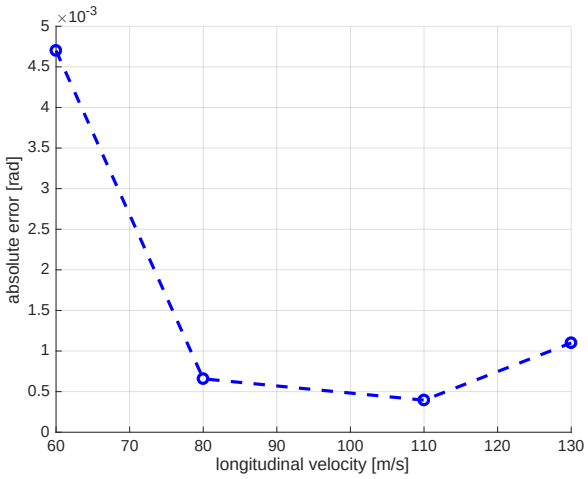


Figure 7.7: Absolute error of the 4-parameters LFT estimator for increasing velocities

Slalom trajectory

At the same time, the IMU has the same predominant role as the previous case at high speeds, allowing a correct estimation, even if the saturation that normally incurs above a certain stress limit cannot be represented, due to the limitations of the single track model.

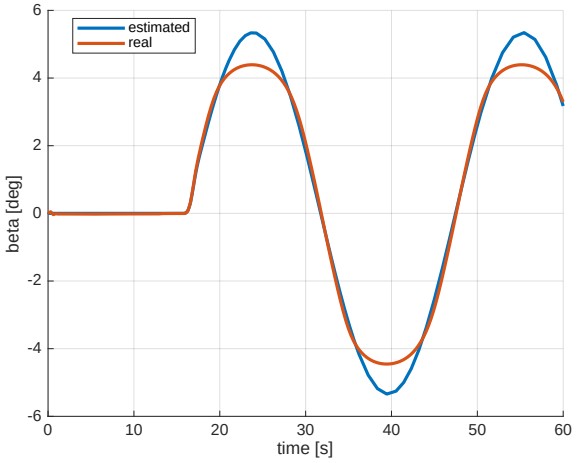
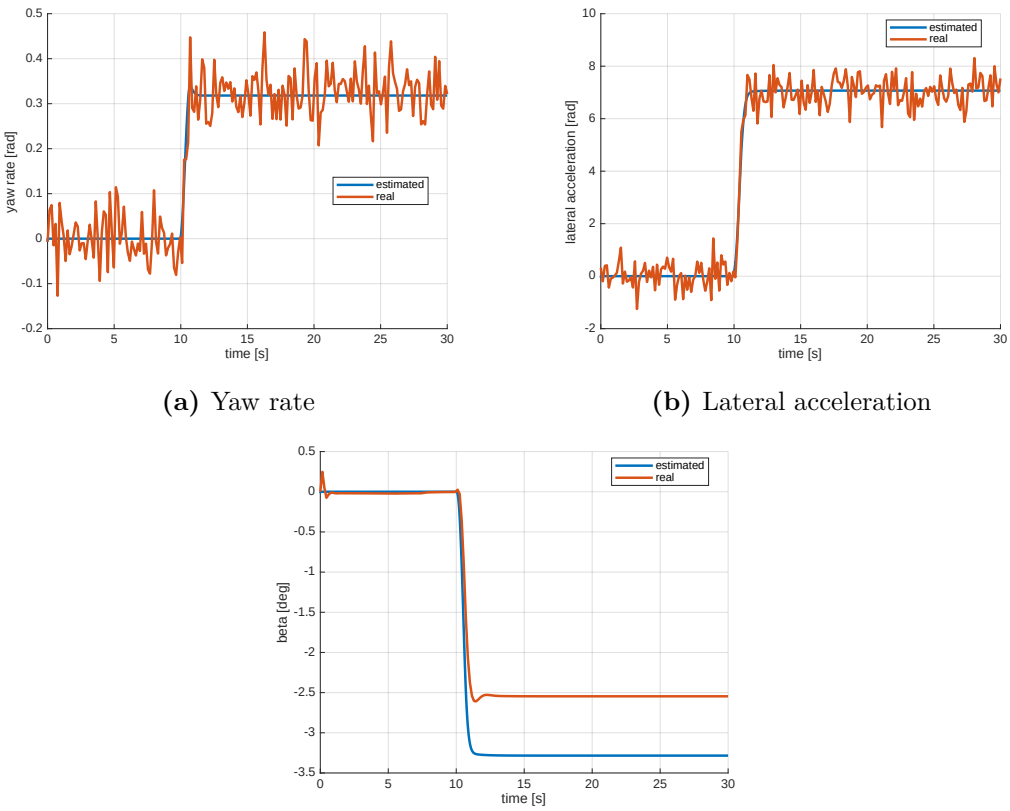


Figure 7.8: Results of the 4-parameters LFT estimator against the slalom trajectory

IMU error sensitivity

From the explained concepts it is clear that the inertial measurements influence significantly the sideslip estimation. This can constitute a problem, as IMU sensors have an intrinsic measurement noise and must be placed and calibrated correctly. Bias errors, introduced by an improper placement or orientation of the sensor, results in a significant error, that cannot be compensated by the position measurements.

On the other hand, the measurement noise associated with these sensors has little effect on the estimation: by introducing a gaussian white noise with a significant variance on the yaw rate and acceleration values, and by repeating the same test in Figure 7.6, we can see that the estimation is not particularly affected, as the tool is built upon a minimization procedure. A smoothing of the data, like the one required by the feedback linearization-based tool, is generally not needed, although it can ease the minimization.



(a) Yaw rate

(b) Lateral acceleration

(c) Sideslip estimation

Figure 7.9: Results of the 4-parameters LFT estimator against noise in IMU measurements

7.3.3 Experimental results

This variant of the LFT-based estimator can be validated against the data set from the Unmanned ground vehicle already considered in the previous cases. The additional ranges required for the identification of the parameters are reported in the table below.

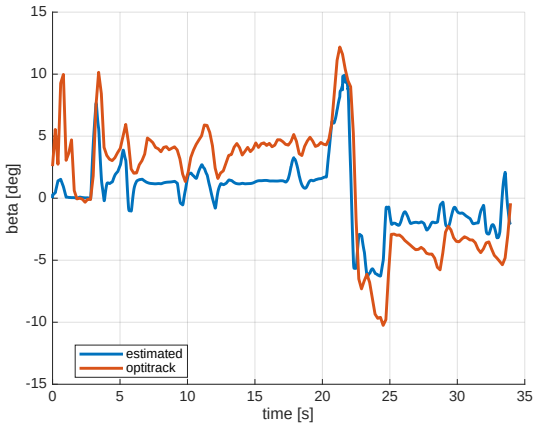
Mass range	m	$1-5$	kg
Inertia range	I_z	$0.01-0.5$	kgm^2

Results provided by the algorithm, reported in Figure 7.10, are similar to the ones obtained by the other variants of the estimator, with the advantage that a precise measurement of the mass and inertia of the vehicle is not necessary.

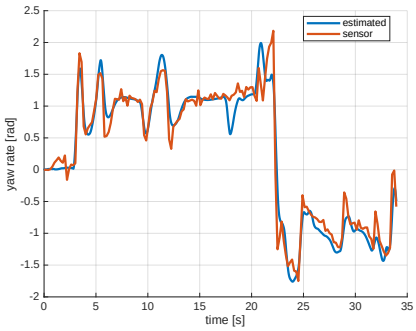
The estimation error is higher for low sideslip angles: there seems to be a threshold value (about $2°$) below which the output tends to zero. Since this issue was not noticed while testing the tool against simulated data sets, it is probably a consequence of the tuning of the IMU platform: a wrong orientation can lead to significant errors. Nonetheless, this problem is solvable by a proper setup of the sensors.

TIC values, calculated through equation (4.31) and reported in the table below, show that this tool is the best among the ones considered when dealing with the interpolation of available inputs, a characteristic expected, since it is the most flexible, but not corresponding to a better ability in estimating β.

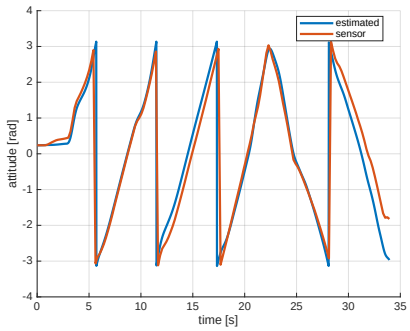
	position and IMU-based LFT estimator with 4 uncertain parameters	position and IMU-based LFT estimator with 2 uncertain parameters	position-based LFT estimator with 2 uncertain parameters
TIC_x	0.076978	0.082263	0.084112
TIC_y	0.018328	0.06969	0.071208
TIC_ω	0.022698	0.040071	-
TIC_{a_y}	0.085612	0.096856	-



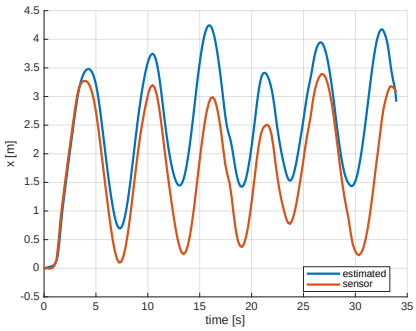
(a) Slip angle



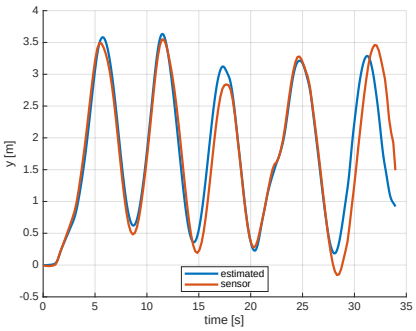
(b) Yaw rate



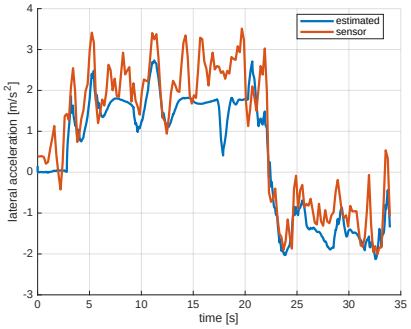
(c) Attitude



(d) Position (x axis)



(e) Position (y axis)



(f) Lateral acceleration

Figure 7.10: Results of the 4-parameters LFT estimator against data from a UGV

7.3.4 Conclusions

The considered variant of the tool is probably the most flexible, as all needed parameters are identified online and chosen to reflect the dynamics of the considered vehicle.

Sensitivity to IMU measurements can constitute a problem, that must be solved through a proper tuning of the sensor. The LFT-based estimator with only two uncertain parameters has proven to be more effective when dealing with real data sets, as its precision is less affected by measurement noise. A solution for increasing the robustness consists in choosing a narrower identification range for one or more parameters, or constraining one parameter to a static value: this operation can be performed without changing the underlying math, but simply by tuning the identification algorithm.

Unfortunately, the parameter identification procedure is slow and critical, and a wrong choice of initial state values and parameters can lead to an ill-formed problem, at least with the considered minimization method (Matlab-implemented trust region algorithm). The actual Matlab implementation of the tool is fit as an offline tool, while it can hardly be used as an on-board sideslip estimator. Nonetheless, a future version, focused on performance, could work as expected.

7.4 Repeated identification

7.4.1 Working principle

Until now the two steps of the LFT-based estimation have been performed once for each data set, and on the same timespan. A single identification has followed a single simulation of the model. This scenario is critical for two reasons:

- Cornering stiffnesses depend on the considered terrain, so it is not appropriate to estimate them once.
- Cornering stiffnesses cannot be considered as constants, as they slowly decrease while β increases, up to a saturation point.

A considerable improvement consists in splitting the estimation procedure from the parameter identification procedure, that is repeated with a fixed period, using the samples gathered after its last run, as schematized in Figure 7.11. Efficiency could be improved by considering only the samples gathered in a reduced time window, but this collides with the main limitation of this approach: the identification requires the entire initial condition vector, that in case of the considered model consists in values of v, r, ψ, x, y at the initial time sample. These can be partially obtained from available sensors, except from the lateral velocity v (it can be obtained through the integration of the lateral acceleration, but this operation was discarded not to introduce a drift error). The missing value can be computed from a different measurement system or, more easily, extracted from the final state vector of a previous estimation, that must end at the exact time instant in which the next estimation starts, making necessary the consideration of sequential samples.

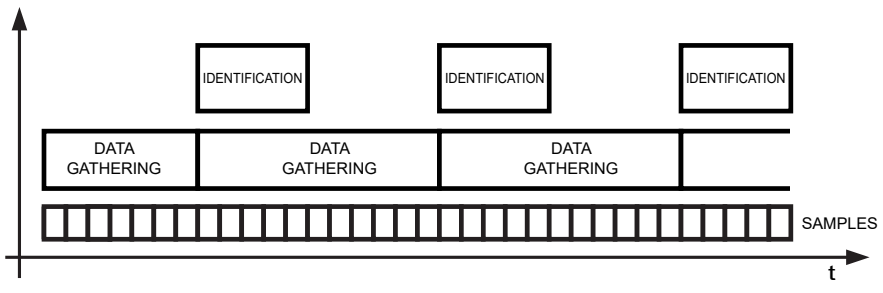


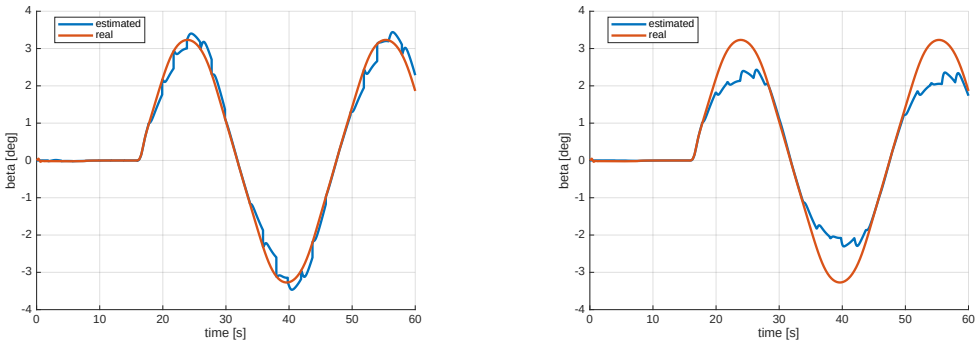
Figure 7.11: Working principle of the repeated LFT identification procedure

This approach can be applied to all the considered variants of the formulation: for simplicity reasons, it is demonstrated with the formulation that uses position and IMU measurements to identify the cornering stiffnesses.

7.4.2 Simulation results

A first test can be performed against data from the usual simulation (vehicle running at 80 km/h, steering wheel at 45°, slalom trajectory, 60 s), imposing a period of 2 seconds between identifications.

As already anticipated, the weakness of the system is constituted by the choice of the initial state vector of both the identification and the model simulation. By filling x_0 with ideal measurements, results show that this approach is able to estimate almost every situation, even the one not trackable by the considered model, like in case of not constant cornering stiffness values due to increasing solicitations, as shown in Figure 7.12.



(a) Estimation with ideal initial state vector

(b) Estimation with available initial state vector

Figure 7.12: Results of the repeated LFT identification procedure

The problem arises when x_0 is filled with real measurements from available sensors and previous simulations of the model (as mentioned early, measurements are not enough to fill the entire vector): the system is unable to track the non-linearities that arise at high speeds, in a way similar to the one of the single identification-based estimator.

Furthermore, the estimator performs worse than the others, as multiple identifications are unable to account for the entire behavior of the vehicle, and properly eliminate the error. This can be mitigated by increasing the period between identifications. A comparison of results obtained with different time periods is reported in Figure 7.13: by using a longer timespan it is possible to obtain more precise estimations, but since the choice of the initial state is critical, successive estimations perform worse than the previous ones.

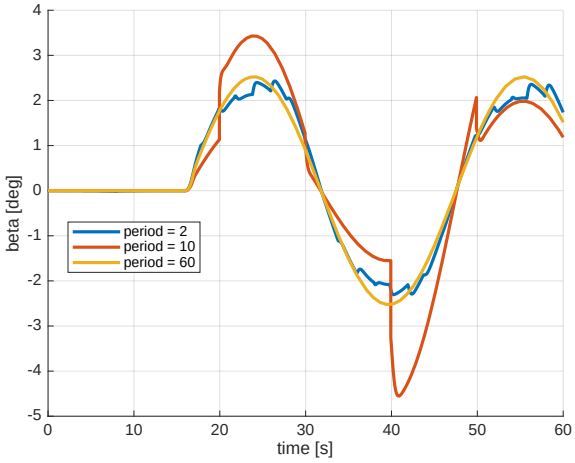


Figure 7.13: Results of the repeated LFT identification procedure for increasing periods

7.4.3 Conclusions

A repeated identification procedure is necessary to use a LFT-estimator variant as a real time beta sensor. Unfortunately, the precision of the results depend on the robustness of the previous identifications.

A useful feature that could be added to an implementation consists in a reset procedure, similar to the one developed for the feedback linearization-based estimator and described in Section 5.6, that automatically detects improper estimations and reset the state values. Repeated testings show that this approach allows estimators with limited stability to perform in every context.

7.5 Future improvements

The flexibility of the LFT approach for parameter identification allows to improve the performance of the base sideslip estimator, by adding arbitrary comparison measurements or uncertain parameters. Moreover, a real world usage is is possible by implementing a repeated identification procedure, even if less precise than the single estimation one.

Nonetheless, the considered tool requires significantly more computational power than simpler systems, like the feedback linearization-based one, and the error convergence depends on a series of technical parameters that must be tuned accurately. Improvements in this field have been represented by the addition of a constrained search to the toolbox, but a more accurate study needs to be performed around the stability of the minimization algorithm.

Apart from the problems linked to the identification procedure, the implemented system works well in standard situations, while it seldom works in non-adherence conditions like drifting, due to the limitations of the considered relationship between the lateral force and the slip angle. A considerable improvement would consist in the addition of a full tyre model, a path already followed in similar works. This would result additional computational requirements, additional parameters to identify, and probably to additional required measurements, that can come from additional inertial sensors placed in multiple parts of the vehicle.

Another technique for obtaining a realistic sideslip estimation could consist in using together multiple approaches, weighting differently their contribution according to the speed: as already seen, the LFT-based estimator behaves well at high speeds, regardless of static parameters, so it could be accompanied by a another algorithm, that is considered at low speeds.

Appendix A

Simulation environment

All the estimators presented in this work have been tested against data provided by a virtual environment, whose setup and characteristics are reported in this section. The test workbench has been built through the software Dymola and Simulink.

A.1 Setup

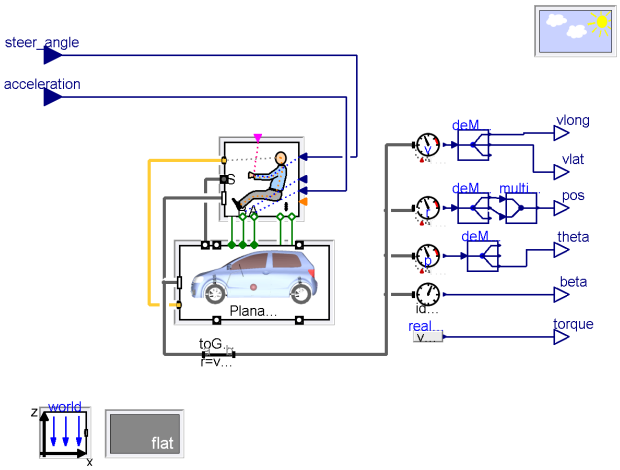


Figure A.1: Dymola part of the simulation environment

Dymola allows to setup a vehicle simulator, able to receive standard driving commands and output kinematic variables. The software is a commercial implementation of the Modelica language, used to build dynamic models, and includes a GUI and specific libraries [2]. The simulator built for the tests, displayed in Figure A.1, is composed by the following components:

- An electric vehicle model, provided by the Vehicle Dynamics Library, that is based on a gearless DC motor replica and four tires, while the lateral dynamics simulation is provided by a modified Pacejka model, similar to the one discussed in Chapter 2, but characterized by saturation limits that arise under high stresses;

- An open-loop automatic driver, a model that allows the direct control of the vehicle through numeric values of acceleration and steer angle. These commands are linked to an input interface.

- A series of ideal sensors on the vehicle chassis, linked to an output interface, that provide measurements of position, yaw angle and slip angle of the center of mass of the vehicle.

- A flat terrain environment, with editable friction coefficient.

The characteristic parameters of the vehicle are shown in the table below. The cornering stiffnesses value C_f and C_r are just estimates, as the underlying model is more complex than the one presented in Chapter 2.

Vehicle mass	m	1100	kg
Vehicle inertia	I_z	1504	kgm^2
Steering gain	G	1/15.9111	-
Distance of front axle from C.O.G.	a	1.00005	m
Distance of rear axle from C.O.G.	b	1.46986	m
Front stiffness range	C_f	59420	N/rad
Rear stiffness range	C_r	40315	N/rad

The simulator can exchange data via Simulink (Matlab block diagram system), which is used to implement the control part, that consists in a cruise control and a series of inputs and outputs, that are made available to each estimator.

The simulation requires the setup of a longitudinal velocity feedback (cruise control), that has to be properly tuned. The use of a complex multibody vehicle model provides a more real behavior with respect to a simpler model (including a not-null slip angle), but an analysis needs to be done to derive a simplified model that can be used to tune an automatic control system. By inspecting the Dymola components, it is possible to discover that the accelerator signal, provided through the input interface, is multiplied by a factor and used as reference in a closed loop that controls motor torque, with its unitary value corresponding to the maximum deliverable torque. If the torque saturation is neglected and the electrical dynamics are much faster than the mechanical ones, we can assume that the model behaves

like this simple dynamic system:

$$\begin{cases} \tau = G_t \cdot accelerator \\ J\frac{d\omega}{dt} + D\omega = \tau \\ v = G_v \cdot \omega \end{cases} \tag{A.1}$$

where ω is the motor speed and v is the longitudinal vehicle velocity. By applying the Laplace transform, it is evident that the the relationship between the accelerator signal and the vehicle velocity is a first order transfer function:

$$G(s) = \frac{v(s)}{accelerator(s)} = \frac{G_t \cdot G_v}{Js + F} = \frac{\mu}{Ts + 1} \tag{A.2}$$

An estimation of all the parameters is useless, since a first order transfer function is characterized only by two parameters (gain μ and time response T). These can be found by feeding the system with a step input, ensuring not to exceed the saturation limits, and passing the output measurements to a linear estimator, that can be built, for instance, through the Grey-box module provided by Matlab, obtaining the values. A comparison of the first order model response with the original system response outlines a very low relative error. It is now possible to draw the Bode Diagram of the transfer function between the vehicle accelerator signal and velocity, as in Figure A.2.

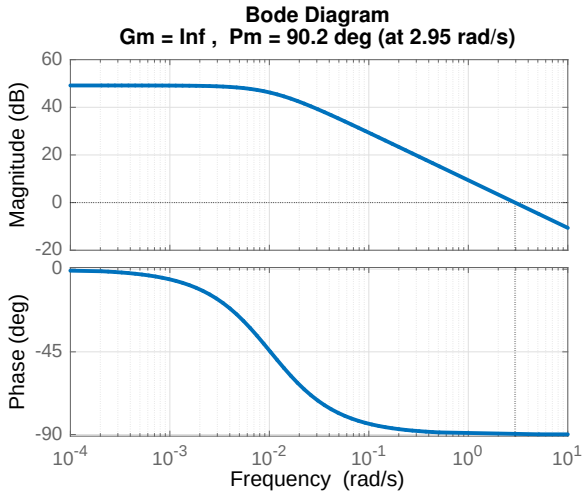


Figure A.2: Transfer function between accelerator and velocity

We can now tune the cruise control, implemented as a PI regulator. We want to enforce a crossover frequency, for instance, of 10 rad/s, limited by the saturation of the motor accelerator. Suitable control parameters can be found by considering the loop transfer function (product of plant G and regulator R):

$$L(s) = R(s) \cdot G(s) = K_p(1 + \frac{1}{sT_i}) \cdot \frac{\mu}{Ts + 1} \tag{A.3}$$

By imposing the crossover frequency and inverting the equation, K_p is obtained, while T_i can be set arbitrarily, at the condition that the associated zero is positioned before the critical frequency. Finally, we have to make sure that the integral action of the regulator does not increase indefinitely due to a saturated acceleration signal: a common technique is called anti-windup and is available in the Simulink PID block. The cruise control can finally be added in the Simulink part of the environment, as shown in Figure A.3.

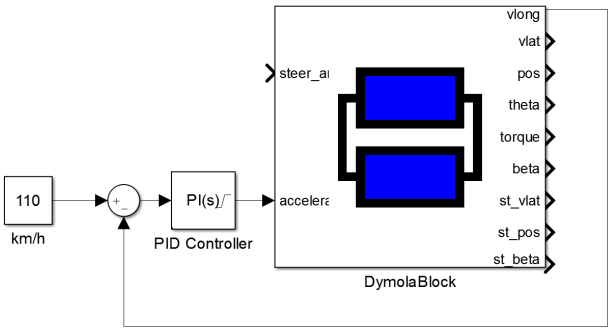


Figure A.3: Simulink part of the simulation environment

A.2 GNSS error model

The experimental testing of all the proposed estimators is mostly dedicated to analyze their robustness against available sensor characteristics. The high sensitivity to measurement accuracy, typical of GNSS-based estimators, has already been introduced in Chapter 2, as the limitations of positioning sensors. A modeling of the latter could therefore allow an evaluation of the real world performance of the tool. Based on the characteristics of GNSS sensors already explained in the state of the art chapter, it is possible to build a model of the expected errors, that can be applied on ideal position measurements.

Various complex models have been developed to simulate the physical behavior of a GNSS sensor [36]. We are not interested in replicating its exact behavior, but only the associated noise: from this consideration, it is possible to build a simpler system. We know that the sensor, despite being affected by a low measurement noise, is highly influenced by the receiving quality of the radio signals, that can change suddenly in presence of obstacles due to the multipath effect [35], causing a shift of several meters in the estimated position.

We can model this behavior by mapping the terrain into a discrete grid, and assigning each pixel a specific signal quality, represented in Figure A.4 as color intensities from black (low) to white (high).

Grid size must not be too big, in order to ensure the presence of homogeneous

zones: previous testings have shown that the value shift is sudden and constant for similar zones.

When the vehicle enters a pixel, the signal quality is read and a corresponding value shift can be applied to the position value.

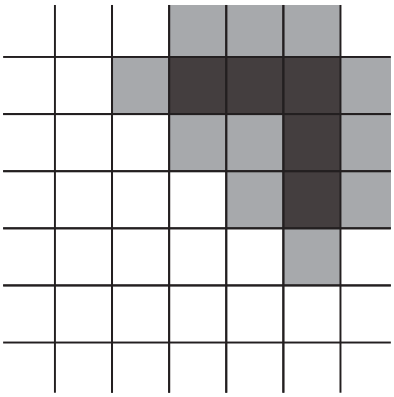


Figure A.4: Example of grid for the GPS error model

An implementation of the model consists in drawing a black and white image in an arbitrary image editing software, with the color intensity of each pixel corresponding to a quality level. The image can be converted through Matlab into a matrix of real numbers. These can be mapped to the terrain and interfaced to Simulink through a 2D lookup table. It is important to use the table in flat mode, an option that forbids the interpolation of values if a position is not at the center of a zone, otherwise shifts are not present in the final result.

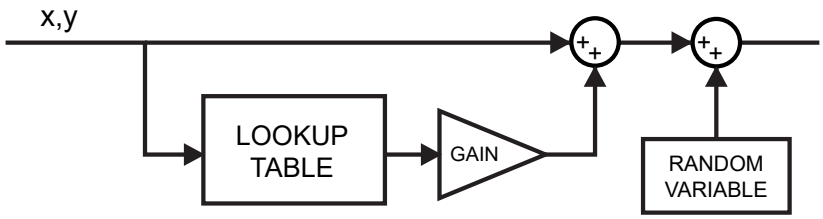


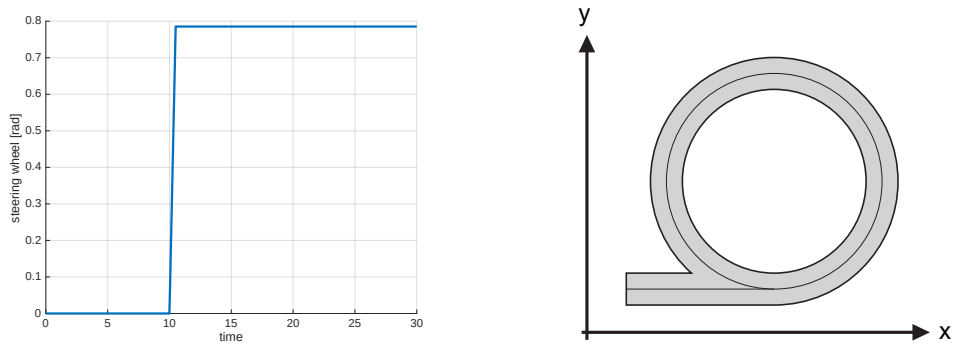
Figure A.5: GPS error model

To obtain a simulated GPS value, the current position is fed into the lookup table, obtaining the quality level. The quality level is then associated to a displacement (in this case, 3 meters corresponds to the lowest quality level), that is added to the position. Finally, a low measurement noise is added to the value. The entire system is represented in Figure A.5.

A.3 Trajectories

A non-null steering signal must be provided to cause a non-null sideslip angle. The following testing trajectories were chosen not only to replicate particular situations, but also to compare the results with the one of similar works.

Roundabout

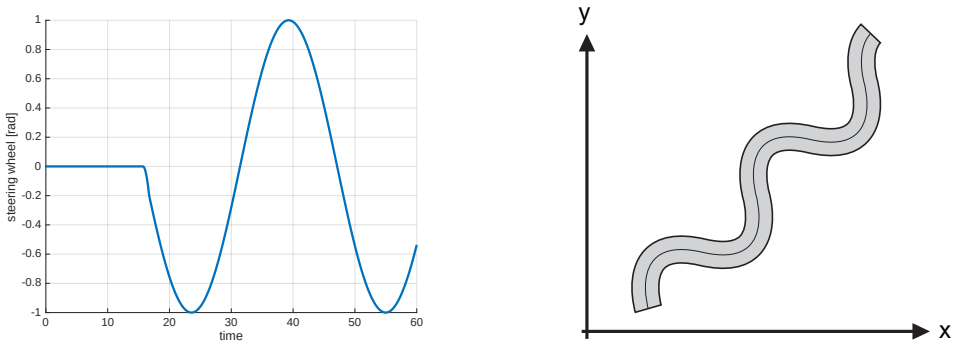


(a) steering signal **(b)** trajectory

Figure A.6: Roundabout trajectory

A step-like steering signal, shown in Figure A.6, corresponds to driving in a roundabout. While the trajectory is not excessively realistic, its use with a sideslip estimator allows a clear evaluation of the system responsiveness for the given speed and steer angle, as it is similar to a step input, typically used for this purpose.

Slalom



(a) steering signal **(b)** trajectory

Figure A.7: Slalom trajectory

A sine wave steering signal, shown in Figure A.7, corresponds to a common driving path that causes a continuous variation of the vehicle forces, making the slip angle continuously varying with time. The ability of an estimator to work properly with such trajectory and high lateral force values depends on its ability to consider the variability in the relationship between the lateral forces and the slip angles.

Appendix B

Unmanned ground vehicle

Experimental tests were performed against data sets obtained from the driving of an unmanned ground vehicle, performed in a laboratory at the DEIB department of Politecnico di Milano. The small car, a standard test asset used for many researches, is equipped with the following components:

- A microcontroller, running the Robot Operating System (ROS), equipped with a Wi-Fi antenna and a radio receiver, and linked to all the on-board sensors;
- An encoder on the left rear wheel, linked to an additional microcontroller (an Arduino) that is in turn linked to the main microcontroller;
- a 9-DOF IMU sensor, providing measurements from an accelerometer, a gyroscope and a magnetometer;
- a positioning sensor compatible with an optitrack system.

Position measurements come from an optitrack system, an optical tracker-based system that is able to detect the vehicle positioning sensor through a set of cameras fixed to the laboratory ceiling. The resolution is greater than most GNSS sensors and is not affected by problems linked to obstructed satellite signals, although the values are sometimes characterized by sudden shifts.

Measurements gathered on the vehicle and by the optitrack system are transmitted through a Wi-Fi network to a ground station, and gathered through the ROS framework. The resulting data set is then recorded in a standard format.

The characteristic parameters of the vehicle, measured or estimated off-line, are reported in the table below.

Vehicle mass	m	1.454	kg
Vehicle inertia	I_z	0.015	kgm^2
Steering gain	G	1	-
Distance of front axle from C.O.G.	a	0.1448	m
Distance of rear axle from C.O.G.	b	0.1144	m

A proper measuring of the vehicle inputs (steering signal and longitudinal velocity) is critical, as these two values are assumed certain and influence sensibly the correctness of the simulation. The other sensors have also to be tuned, but can contain moderate measurement noise without influencing the estimation.

A preliminary processing phase has been introduced for estimation algorithms requiring measurements from the steering wheel, encoder, gyroscope or accelerometer sensors, in which the values are calibrated, cleared and smoothed.

The following procedure has been applied on the available samples:

- The steering signal is time-shifted forward of some milliseconds to compensate the actuation time.
- The longitudinal velocity can normally be obtained through the encoder, but if the latter is unavailable, an alternative procedure to extract the needed quantity consists in projecting the vehicle velocity, tracked by the optitrack, on the vehicle reference frame, through a rotation with the vehicle orientation angle, provided by the optitrack or alternatively by the magnetometer (the latter can be used since the laboratory is a closed and limited space; in other situations its use is discouraged).
- IMU measurements have to be calibrated to refer to the proper directions. A method consists in stopping the vehicle and rotating the measurement vector through a rotation matrix, until the acceleration is concentrated on the z-axis, pointing to the ground. At this point, it is possible to compare values of the on-board magnetometer z-axis, indicating the IMU orientation, with values of the vehicle orientation provided by the optitrack, as they have to differ by a constant offset (due to a different absolute reference). Finally, the resulting rotation matrix can be applied to each IMU sensor.
- The starting orientation of the vehicle has to be set manually through a constant offset, as it is impossible to determine with available sensors.

All values have been fitted with a smoothing spline curve, with a smoothing parameter of 0.995.

Finally, since measurements from different sensors are referred to different time instants, all the data set must be referred to a common time series, that has to be used to interpolate the values. The choice of a time series with a uniform spacing between time instants allows to account for missing measurements.

Appendix C

Visual comparison between all estimation methods

Sideslip estimations provided by all the tools treated in this work, relative to the same data set, obtained by the driving of the unmanned ground vehicle, are reported in Figure C.1. The reference value is provided by the optitrack system, that makes use of the vehicle velocity to compute the quantity of interest.

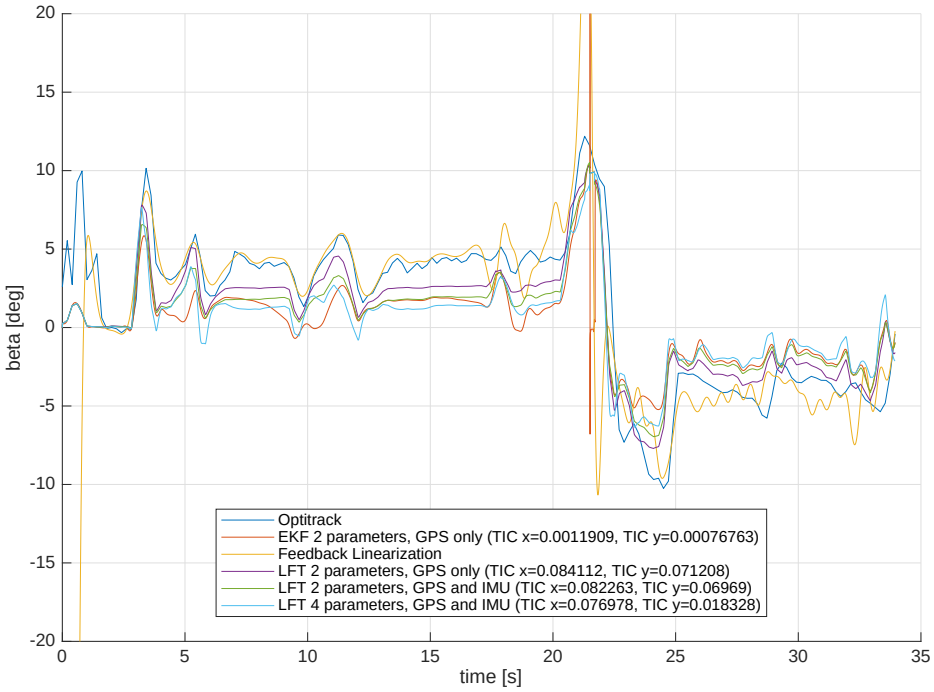


Figure C.1: Visual comparison between all estimation methods

List of Figures

Bibliography

[1] Correvit s-hr datasheet. https://www.kistler.com/?type=669&fid=67238. [cited on page 8]

[2] Dymola user manual volume 1. chapter Getting started with Dymola. April 2016. [cited on page 93]

[3] Matlab r2017b documentation. 2017. [cited on page 61]

[4] F. Aghili and A. Salerno. Driftless 3-d attitude determination and positioning of mobile robots by integration of IMU with two RTK GPSs. *IEEE/ASME Transactions on Mechatronics*, 18:21–31, 02 2013. [cited on pages 9 and 21]

[5] J. Bechtoff, L. Koenig, and R. Isermann. Cornering stiffness and sideslip angle estimation for integrated vehicle dynamics control. *IFAC-PapersOnLine*, 49(11):297 – 304, 2016. 8th IFAC Symposium on Advances in Automotive Control AAC 2016. [cited on page 21]

[6] A. D. Bona. Solutore efficiente per la simulazione di modelli LFT in cascata. Master's thesis, 2014. [cited on pages 57 and 61]

[7] A. D. Bona, G. Ferretti, E. Ficara, and F. Malpei. LFT modelling and identification of anaerobic digestion. *Control Engineering Practice*, 36:1 – 11, 2015. [cited on pages 50, 56, 57, 59, and 62]

[8] M. Bossi. LFT-based nonlinear MPC. Master's thesis, Politecnico di Milano, 2014. [cited on pages 3, 10, 57, 62, and 65]

[9] K. Bryan. Quasi-newton methods notes. Rose-Hulman Institute of Technology. [cited on page 53]

[10] L. Carraro. Identification of the cornering stiffness based on a linear fractional formulation of the vehicle dynamics. Master's thesis, Politecnico di Milano, 2016. [cited on pages 3, 10, 11, 37, 62, and 65]

[11] C. Cheng and D. Cebon. Parameter and state estimation for articulated heavy vehicles. *Vehicle System Dynamics*, 49(1-2):399–418, 2011. [cited on

page 10]

[12] P. Civicioglu. Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, 219(15):8121 – 8144, 2013. [cited on page 50]

[13] F. Donida, C. Romani, F. Casella, and M. Lovera. Integrated modelling and parameter estimation: an LFT-modelica approach. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 8357–8362, Dec 2009. [cited on page 54]

[14] M. Doumiati, A. Victorino, A. Charara, and D. Lechner. Unscented Kalman filter for real-time vehicle lateral tire forces and sideslip angle estimation. In *2009 IEEE Intelligent Vehicles Symposium*, pages 901–906, June 2009. [cited on pages 10 and 21]

[15] G. Erdogan. Tire modeling lecture. University of Minnesota. [cited on page 14]

[16] A. C. H. Francisco Rovira Màs, Qin Zhang. *Mechatronics and Intelligent Systems for Off-road Vehicles*. Springer, 2010. [cited on page 6]

[17] S. Garatti and S. Bittanti. Parameter estimation in the Pacejka's tyre model through the TS method. *IFAC Proceedings Volumes*, 42(10):1304 – 1309, 2009. 15th IFAC Symposium on System Identification. [cited on pages 10, 16, 50, and 67]

[18] S. Garatti and S. Bittanti. A new paradigm for parameter estimation in system modeling. *International Journal of Adaptive Control and Signal Processing*, 27(8):667–687, 2013. [cited on page 11]

[19] H. F. Grip, L. Imsland, T. A. Johansen, J. C. Kalkkuhl, and A. Suissa. Vehicle sideslip estimation: Design, implementation, and experimental validation. *IEEE Control Systems*, 29(5):36–52, Oct 2009. [cited on pages 7, 8, 10, and 21]

[20] D. L. Guillaume Baffet, Ali Charara. Estimation of vehicle sideslip, tire force and wheel cornering stiffness. *Control engineering practice*, 17(11), Nov 2009. [cited on page 10]

[21] M. A. Henson and D. E. Seborg. Nonlinear process control. chapter Feedback Linearizing Control, pages 149–231. Prentice-Hall, 1997. [cited on page 37]

[22] K. Hsu, T. Vincent, G. Wolodkin, S. Rangan, and K. Poolla. An LFT approach to parameter estimation. *Automatica*, 44(12):3087 – 3092, 2008. [cited on page 56]

[23] ISO 15037-1:2006. Road vehicles – vehicle dynamics test methods. Standard, International Organization for Standardization. [cited on page 8]

[24] C. Lamy, J. Caroux, M. Basset, G.-L. Gissinger, D. Poli, and P. Romieu. Comparison of optical and GPS based tire slip angle estimation. *IFAC Proceedings Volumes*, 40(10):41 – 46, 2007. 5th IFAC Symposium on Advances in Automotive Control. [cited on pages 3, 8, 9, and 21]

[25] L. Ljung. Prediction error estimation methods. *Circuits, Systems and Signal Processing*, 21(1):11–21, Jan 2002. [cited on page 50]

[26] L. Ljung. Approaches to identification of nonlinear systems. In *Proceedings of the 29th Chinese Control Conference*, pages 1–5, July 2010. [cited on page 50]

[27] A. D. Luca, G. Oriolo, and M. Vendittelli. Stabilization of the unicycle via dynamic feedback linearization. *IFAC Proceedings Volumes*, 33(27):687 – 692, 2000. 6th IFAC Symposium on Robot Control (SYROCO 2000), Vienna, Austria, 21-23 September 2000. [cited on pages 3, 9, 39, and 40]

[28] A. K. Luca De Ambroggi. Automotive semiconductor market tracker, q2 2017. https://technology.ihs.com/589527/automotive-semiconductor-market-tracker-q2-2017. Accessed: 2017-10-15. [cited on page 1]

[29] G. Mastinu and M. Ploechl. *Road and Off-road Vehicle System Dynamics Handbook*. Hoepli, 2013. [cited on pages 5 and 17]

[30] G. Morrison and D. Cebon. Sideslip estimation for articulated heavy vehicles at the limits of adhesion. *Vehicle System Dynamics*, 54(11):1601–1628, 2016. [cited on page 10]

[31] H. Pacejka. *Tyre and Vehicle Dynamics*. Butterworth-Heinemann, 2006. [cited on pages 7, 10, 14, and 17]

[32] P. Pettersson. Estimation of vehicle lateral velocity. Master's thesis, Lund University, 2008. [cited on pages 3 and 10]

[33] U. Qureshi and F. Golnaraghi. An algorithm for the in-field calibration of a MEMS IMU. *IEEE Sensors Journal*, 17(22):7479–7486, Nov 2017. [cited on page 13]

[34] R. Rajamani. *Vehicle Dynamics and Control*. Springer, 2011. [cited on pages 1 and 5]

[35] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice-Hall, 2001. [cited on page 96]

[36] M. B. Riccardo Fesi. Sistema di navigazione per un all-terrain vehicle basato su tecniche di sensor fusion. Master's thesis, Politecnico di Milano, 2014. [cited on pages 3, 12, and 96]

[37] J. Ryu and J. Gerdes. Vehicle sideslip and road bank angle estimation using GPS. 25:347–359, 07 2004. [cited on pages 3, 9, and 35]

[38] M. Spaliviero. Control of an autonomous all-terrain vehicle with model predictive control. Master's thesis, Politecnico di Milano, 2016. [cited on page 37]

[39] R. F. Stengel. *Optimal Control and Estimation*. Dover Publications, Inc., 1994. [cited on pages 21, 24, 26, 50, 51, and 64]

[40] A. J. Tuononen. Vehicle lateral state estimation based on measured tyre forces. In *Sensors*, 2009. [cited on pages 3 and 10]

[41] C. Voser, R. Y. Hindiyeh, and J. C. Gerdes. Analysis and control of high sideslip manoeuvres. *Vehicle System Dynamics*, 48(sup1):317–336, 2010. [cited on page 21]

[42] J. H. Yoon and H. Peng. Robust vehicle sideslip angle estimation through a disturbance rejection filter that integrates a magnetometer with GPS. *IEEE Transactions on Intelligent Transportation Systems*, 15(1):191–204, Feb 2014. [cited on page 13]