POLITECNICO DI MILANO
Master of Science Program in Automation and Control Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

CONTROL OF IN-WHEEL MOTORS FOR AN OUTDOOR SKID-STEERING ROBOT

Supervisor: Prof. Luca Bascetta

A thesis submitted by:

Orlando Enrique Díaz Pompa, 833956
Iván de Jesús Iguarán Suárez, 833875

in partial fulfillment of the requirements
for the degree of Master of Science.

Academic Year 2015-2016

For my family,
who kept me strong everyday;
ab imo pectore.

-Iván Iguarán-

To my grandfather Luis,
because is the second degree that
I would have loved to share with you.
To my uncle Alberto,
because you were right:
there is nothing like travelling.

-Orlando Díaz-

Summary

The purpose of this thesis is to analyze the effects of non-flat surfaces and non-ideal tire models on the motor control of a 4 in-wheel skid-steering mobile robot. The reason for this research is to design an outdoor vehicle capable of working on harsh conditions.

The thesis is composed of 6 chapters, each of them dealing with different aspects of the modeling, comparison and implementation. Chapter 1 contains the problem statement, it briefly explains the issue that we are facing. Chapter 2 details the model of the system starting from the most basic concepts. This chapter is divided into the kinematic description of the skid-steering theory, the dynamics behavior of a mobile robot, the modeling of a DC brushless motor and a trajectory generator to plan a realistic path. Chapter 3 includes an explanation of the control architectures that we have considered and the final choices that we have made.

Chapter 4 focuses in the simulation and comparison of the system under different conditions. It contains all the tests necessary to verify the modeling stated in Chapters 2 and 3.

Chapter 5 deals with the implementation of a single motor control and the experimental results. It starts by introducing the Field Oriented Control theory, which is applied in the software to accomplish the control. Finally, a set of tests were conducted in order to validate the control loops.

The results show that the tires introduce dynamical transients that must be taken into consideration. The motors seems to drive the vehicle properly but it must be taken into account that they can overheat if they are running too much time in steep slopes. It can be concluded that more information is necessary to estimate the physical limits of the vehicle.

Finally, more research must be conducted to generate a trajectory planning that optimizes the motor's performance. It was shown that a proper selection of the timing law allows the system to run in steep slopes without saturating the motors' currents, reducing the power consumption and extending its autonomy.

Sommario

L'obiettivo della tesi è analizzare gli effetti di superfici non piane e modelli con pneumatici non ideali sul controllo del motore di un 4 in-wheel skid-steering robot. Lo scopo di questa ricerca è progettare un veicolo da esterno capace di funzionare in condizioni difficili.

La tesi è composta di 6 capitoli, ognuno dei quali si occupa di diversi aspetti della modellizzazione, del confronto e dell'implementazione. Il capitolo 1 contiene l'esposizione del problema e spiega l'argomento che stiamo affrontando. Il capitolo 2 analizza il modello del sistema partendo dai concetti basilari. Questo capitolo è diviso in una descrizione cinematica della teoria dello *skid-steering*, la dinamica del comportamento di un robot, la modellizzazione di un motore DC brushless e un generatore di traiettorie per stabilire un percorso realistico. Il capitolo 3 include la spiegazione dell'architettura di controllo che abbiamo considerato e le scelte finali che abbiamo fatto.

il capitolo 4 si concentra sulla simulazione e la comparazione del sistema sotto diverse condizioni. Contiene tutti i test necessari per verificare il modello definito nei capitoli 2 e 3. Il capitolo 5 si occupa dell'implementazione del controllo di un singolo motore e dei risultati sperimentali. Si apre introducendo la teoria del Controllo a Orientamento di Campo, che è applicata nel software per ottenere il controllo. Infine, una serie di test sono stati condotti per dare validità all'anello di controllo.

I risultati mostrano che i pneumatici introducono transitori dinamici che devono essere presi in considerazione. Il motore sembra condurre il veicolo in maniera appropriata, ma bisogna tenere in considerazione che possono surriscaldarsi se si muovono per troppo tempo su pendenze ripide. Si può concludere che sono necessarie più informazioni per stimare i limiti fisici del veicolo. Infine, dovrà essere condotta altra ricerca per generare una traiettoria che ottimizzi la performance del motore. Si è evidenziato che un'appropriata selezione della legge oraria permette al sistema di muoversi su pendenze ripide senza saturare le correnti del motore, riducendo il consumo di energia ed allungandone l'autonomia.

Acknowledgments

I would like first to thank God who keeps me strong to face the life's challenges, to my thesis partner Orlando Díaz Pompa, without him anything of this could be possible, to our thesis supervisor Prof. Luca Bascetta who gave us his advice and knowledge for one year and guided us in the right direction. Finally, I would like to thank my family who provided me support and encouragement to accomplish my goals and my friends for being there when I need them.

-Iván de Jesús Iguarán Suárez-

I would like to thank the university and the excellent professors from which I have learned so much in the past two years, specially to my tutor Professor Luca Bascetta, because with him I have learned many things about what is written in this book. Also, thanks to all my friends for the company in this adventure of mine. Thanks to my Colleague Iván Iguarán, with whom we have dealt with so many problems and solutions. Finally but not least, thank you very much to my family and especially to my mother, my father and my brothers, because you are the pillars in which I sustain myself every day.

-Orlando Enrique Díaz Pompa-

Contents

List of Figures

List of Tables

Introduction

The field of mobile robots is an area that has been greatly expanded in the recent years. However, they remained for many years within the universities' walls due to their prohibitive costs and the lack of computational power. It was not until the NASA started the *Mars Pathfinder Project* and in particular, when the rover *Sojourner* landed onto the red planet and explored it for about three months, that the mobile robots began to be seen as safer and cost effective alternatives to replace the human working on high-risk jobs or tasks that exceed our physical capacities.

The agricultural field has endless applications for the mobile robots, however, it is a challenging area. The ever changing environment and the unpredictable weather conditions cause the need to develop a robust platform capable to adapt to such situations. Unlike an indoor mobile robot, an outdoor one needs to take into account the effect of real situations that could affect its behavior, for instance, changing terrain conditions and variable slope. That is why it becomes necessary to conduct a study to know how this variables can affect the vehicle and in particular their motor drives.

In order to answer that question, the dynamic and kinematic model of a *Skid-Steering Robot* and the one of a DC brushless motor are developed and coupled together. A simulated trajectory around a vineyard with different slopes is designed by using a trajectory generator capable to plan paths and timing laws to control the speed of the robot at each point along the route. This is done to evaluate the behavior of the robot and specifically of its motors under specific conditions. Also, two different tire models are considered: a first linear approximation using the Coulomb friction equation and a more realistic model using the *Pacejka magic formula.* Their results are compared and the conclusions about their response for different terrain situations are stated.

Finally, a DC brushless motor controlled by *Field Oriented Control (FOC)* is implemented on an electronic board as a first step in the assembly of the mobile robot in order to validate the control loops. The idea was to test that the current and speed loops had the same behavior as observed during simulations in MATLAB/Simulink. Thus, these experiments could be a first validation to proceed with the assembly process, allowing more complex tests to evaluate the performance of motors in specific situations.

The results indicate that the forces applied to the vehicle are not negligible, the nonlinear modeling of the wheels shows dynamic transients that do not appear using the Coulomb friction equation. These results are useful to study whether the electrical drives have the force required to move the vehicle in demanding environments. The obtained results will also give us an insight about the theoretical autonomy of the system with the implemented batteries. It is verified that the tests in Simulink are valid, since in the experiments implementing the electronic board, the responses are correlated. Recommendations are given regarding which procedure is the most advisable to follow, it is necessary a characterization of the motor to evaluate not only the control loops, but also the electrical performance.

The problem statement is given in Chapter 1, it describes the issues that need to be addressed by the thesis. After in Chapter 2, it will be introduced the kinematics and dynamics of a skid-steering vehicle. In addition, a basic model of a DC brushless motor and its controllers will be coupled to the mobile robot model. It will be also presented a simple trajectory generator capable to simulate paths resembling real movement through different paths. All necessary variables to understand the operation of the robot are shown.

Chapter 3 introduces the control architectures that were considered during the design and their behavior when they are implemented. It will be shown a comparison of their performances by using their sensitivity transfer functions and their step responses. It also explains the benefits of the drivers that were selected and why these are suitable for this specific system.

In Chapter 4, several simulations will be conducted, some of them regarding the control loops and some others the behavior of the vehicle running through two different trajectories. Also the two different tire models will be compared in order to conclude about the forces acting on the vehicle.

Finally, in Chapter 5, a software to control a DC brushless drive using *FOC* is implemented in a commercial developing platform from the brand Renesas. It contains the hardware description, the used libraries, the algorithm and several experiments in order to validate the control system. Additionally, this illustrates real systems being implemented for both the development of the robot and for experiments.

Chapter 1

Problem Statement

The development of technologies that allow the execution of works that are beyond human capabilities has increased in recent years. Because of this, it is necessary to optimize the design of devices that perform these activities, since the efficiency during the implementation will depend on them. Thus, for the users it is essential the continuous improvement because the productivity and costs involved in a specific task vary depending on the performance. A specific case of these devices are the outdoor robots.

Robots for outdoor works require special design considerations, since environmental conditions add variables with a direct effect on their behavior. Then, it is clear that the vehicle robustness is a key point to deal with these variations and to finally succeed in the implementation process. Similarly, reducing the complexity of the mechanical system helps to make simpler the design, allowing all efforts to focus on optimizing motors performance.

Building an outdoor robot with in-wheel motors removes the need for transmission and steering systems, providing the ability to automate these functions through predetermined trajectories. However, this process requires the modeling and design of motor controllers to be sufficiently effective to avoid problems when the robot is in use. It is clear that for the proper implementation of this design methodology, the motor control system is the basis to get satisfactory results comparable with the expected scope.

Motors and their controllers require a performance study that must be carried out by pushing to the limit variables that affect the system dynamics. Figure 1.1 shows a vineyard, which is an area with positive and negative slopes, curves of almost 180 degrees and, in addition, it usually has changes in the friction coefficient of the ground, since the environmental variations influence its value. This is the kind of paths that could provide relevant information to make a preliminary assessment of the design.

Figure 1.1: Vineyard in a mountaing with slope variations.

The design of a *4 in-wheel skid-steering robot* requires as a first step modeling the entire system from the dynamic and kinematic point of view. After that, the following is the assessment of controllers which require a validation process. Although it is possible to perform simulations on models, it is not advisable to go to the assembly process without having first implemented and validated the critical system components. These are the motors and they imply an analysis of the drivers.

About the motors, it is important to know the autonomy when the vehicle is driving a certain path, so it is common to perform a consumption test for specific trajectories. Anyone interested in using a robot like this, will have a special interest not only in the performance, so it is advisable to mention the durability of the components when the robot is working in conditions that can overheat the motors and any other relevant information about the electrical behavior.

Finally, it is required to make an experimental evaluation to the drivers because the ability of the robot to follow a given trajectory automatically depends on them. Control over the electrical and mechanical dynamics will result in variables that can be manipulated so that the vehicle reacts not only to direction changes, but also to changes in the slope and in the friction of the ground. For our purpose, the control variables are the robot speed and the electric current. Through these parameters, a specific torque is applied to the wheels, resulting in a displacement of a magnitude and direction according to the requirements.

Chapter 2

Model of a Skid-Steering Robot

2.1 Introduction

A typical vehicle configuration is composed by a central motor, which transmits its power to the wheels and a steering system based on a set of bars that change the orientation of the front wheels to vary the direction of the movement. In this chapter we will follow a step by step procedure to develop the dynamic model of a different system, in which four motors, each one coupled to each wheel, give power to drive a vehicle. The skid-steering method is used to steer the system, which is based on the fact that the orientation can be changed by modifying the velocity difference between each side of the vehicle. Also, we will develop a model to study the effect of the gravity force on the vehicle when it rolls on a non-flat surface.

The steps will be divided in two major parts: kinematics and dynamics. In the first part we will do the following:

1. It will be studied the relationship between the local vehicle reference frame and the global inertial reference frame, from which we will be able to obtain the transformation matrices to project the coordinates, velocities and acceleration from one frame to the other and vice versa.

2. The skid-steering method will be presented together with the relationship between vehicle velocities on the local reference frame and the angular velocities of the wheels.

While on the dynamics the following will be introduced:

1. The dynamic model of the motors coupled to the wheels will be shown,

together with its control system. We will be able to obtain the output torques that will drive the vehicle.

2. The equation of motion of the vehicle will be implemented, taking as inputs the motors' torques and giving as outputs the real velocities of the vehicle on the local reference. Also, the real angular velocities of the wheels will be used as the feedback signal of the motor control system. The effect of the gravity will be added to the dynamics so as to simulate non-flat surfaces.

2.2 Kinematics

2.2.1 Relationship Between Local and Global Frame

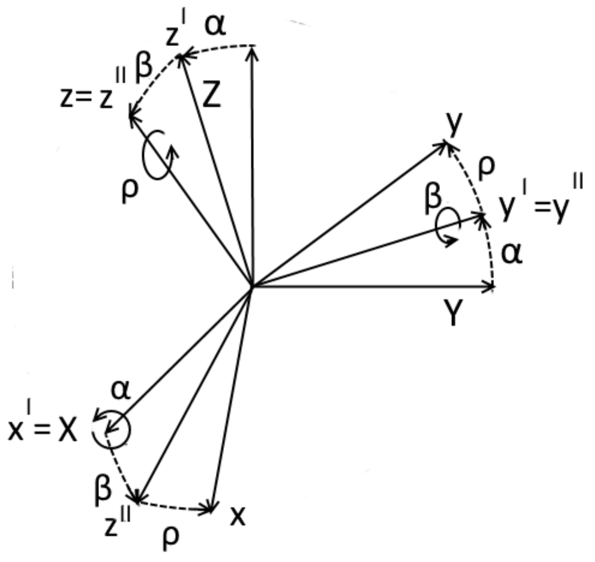


Figure 2.1: Here we can see the rotation of the frame L. First rotating in X, then in y^I *and finally in* z^{II}.

Lets denote two different reference frames: a local reference coordinate $L(x, y, z)$ attached to the center of mass (COM) of the vehicle and a global inertial system $G(X, Y, Z)$ with its origin in a generic point in the space. When the vehicle moves, the location of L changes with respect to G, therefore, it is necessary to obtain a way to calculate its position.

In general, the orientation of a rigid body can be done through many approaches. However, we will use the Cardan angles, which are very used in the aeronautical and nautical industry because the angles are directly related to the roll, pitch and yaw of a ship. In this orientation system, three successive rotations around 3 axes define the position of L. Depending on the order of rotation, the final configuration of the vehicle changes, that is why is very important to follow always the same order.

The procedure can be seen in Figure 2.1 and will be done as follows [18]: in the initial state, the reference L coincides with G. A first rotation around X through the angle α moves the reference G to an intermediate orientation L^I. Then, a second rotation of L^I through angle β around y^I change its position to another intermediate position L^{II}. A final rotation around z^{II} using the angle ρ brings the body reference to its final orientation L. This order of rotation is called $x-y-z$ and the transformation matrices are the following:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) \\ 0 & -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{2.1}$$

$$R_y = \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) \\ 0 & 1 & 0 \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \tag{2.2}$$

$$R_z = \begin{bmatrix} \cos(\rho) & \sin(\rho) & 0 \\ -\sin(\rho) & \cos(\rho) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

We can now denote the transformation from G to L with the rotation matrix:

$$R_{L-G} = R_z R_y R_x \tag{2.4}$$

and from L to G is simply the inverse matrix. Due to the fact that (2.1), (2.2) and (2.3) are orthogonal matrices, the inverse matrix coincides with the transpose, so:

$$R_{G-L} = R_{L-G}^{-1} = R_{L-G}^{T} \tag{2.5}$$

under the assumption that the vehicle moves on a planar surface ($\alpha = 0$ and $\beta = 0$), L will rotate only around the z-axis with angular velocity $\dot{\psi}$. Therefore, the rotation matrices can be simplified to:

$$\begin{bmatrix} x \\ y \\ \psi \end{bmatrix} = \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ -\sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \psi \end{bmatrix} \tag{2.6}$$

$$\begin{bmatrix} X \\ Y \\ \psi \end{bmatrix} = \begin{bmatrix} \cos(\rho) & -\sin(\rho) \\ \sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \psi \end{bmatrix} \tag{2.7}$$

with $\dot{\rho} = \dot{\psi}$, $\rho = \psi$. The vector of coordinates and its derivatives $q = [X \quad Y \quad \psi]^T$ defines the position, velocities and accelerations of the vehicle in the frame G.

By deriving Equation (2.7) we can obtain the relationship from L to G for the velocities obtaining:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} \dot{x} - \dot{\psi} y \\ \dot{y} + \dot{\psi} x \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{2.8}$$

Where v_x and v_y denotes the absolute linear velocities in L. The rotation matrix for planar movement from L to G is :

$$R_{G-L_{planar}} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \tag{2.9}$$

and from the global reference to the local one is simply:

$$R_{L-G_{planar}} = R^T_{G-L_{planar}} \tag{2.10}$$

The acceleration relationship is obtained by deriving again (2.8):

$$\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} \dot{v_x} - \dot{\psi} v_y \\ \dot{v_y} + \dot{\psi} v_x \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \ddot{\psi} \end{bmatrix} \tag{2.11}$$

where a_x and a_y are the absolute linear accelerations in the frame L.

In summary, the rotation matrices (2.9) and (2.10) allow us to transform the coordinates and its derivatives from one reference to the other one.

2.2.2 Skid Steering

Introduction

The most widely used steering system nowadays is the Ackerman Steering, where a complex mechanical system changes the orientation of the wheels with respect to the vehicle to allow turning. Although it allows a good controllability, it lacks of manoeuvrability and mechanical simplicity. Differential steering, instead, uses two wheels, one on each side of the vehicle, and achieves the steering by controlling their angular rotation. It provides high manoeuvrability with zero turning radius but it is mostly used indoors due to the fact that it does not have a strong traction and has low mobility.

In skid-steering, the principle is similar to the differential steering but each side has more than one wheel. It is a system that provides high maneuverability and high response while keeping strong traction and high mobility with the additional advantage of having a robust mechanical structure. Though, it is necessary to keep synchronized the wheels on the same side [21].

Skid Steering Theory

To elaborate this theory, we will use the same approach followed in [9]. Lets assume that the robot moves on a plane surface with the local reference system L(x,y,z) attached to its center of mass (COM) with 'x' pointing to the front, 'y' to the left side and 'z' going up. An inertial reference frame G(X,Y,Z) is also set with its origin in an arbitrary point in the plane. Now, suppose that the vehicle's COM moves with absolute linear velocity in L $\vec{v} = [\ v_x\ v_y\ 0]^T$ and angular velocity $\vec{\omega} = [\ 0\ 0\ \dot{\psi}]^T$. Consider also 'a' and 'b' as the distance between the front and rear wheelbases to the COM respectively and 'w' the displacement between the left and right wheels, as shown in Figure 2.2.

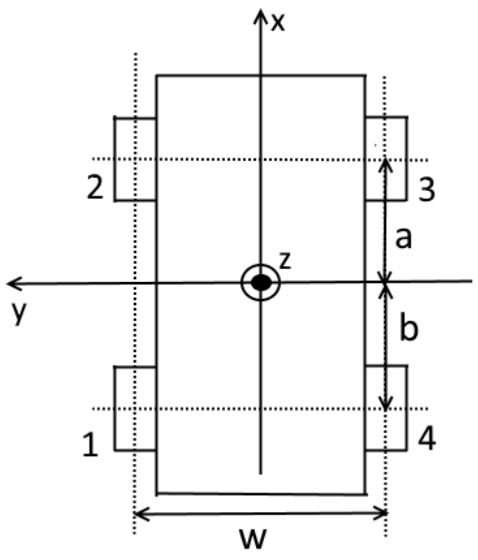


Figure 2.2: Dimensions of the vehicle, measured from the center of the wheel.

We enumerate the wheels, starting from the rear left one that is denoted with a 1 and the rest increasing turning clockwise. Now, we can denote the velocity of each wheel with $\vec{v_i}$, $i = 1, 2, 3, 4$ and its projection onto the reference L with $\vec{v_i} = [v_{ix}\ v_{iy}]^T$. By neglecting its width, we can consider P_i as the contact point between the wheel and the surface, as shown in Figure 2.3.

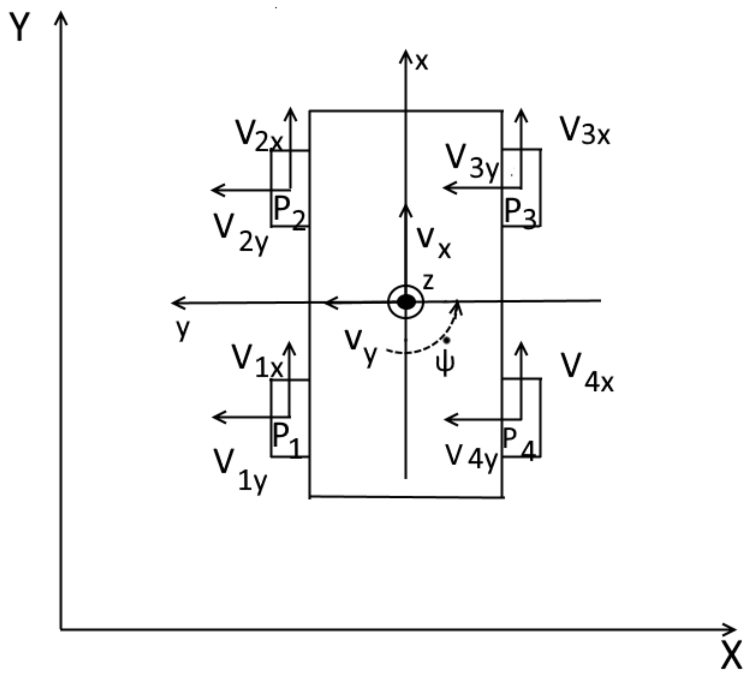


Figure 2.3: The velocities of the COM and the wheels with respect to the local and global reference frame.

In a three dimensional motion, the velocity of the center of the wheel is not aligned with its plane. Therefore, it is necessary to define longitudinal slip of the wheel as [5]:

$$\epsilon_l = \frac{v_x - r_i w_i}{|v_i|} \tag{2.12}$$

and the transversal slip component is:

$$\epsilon_t = \frac{v_y}{|v_i|} \tag{2.13}$$

r_i is distance from the contact point P_i to the center of the wheel, and w_i is its angular velocity.

By assuming that the longitudinal slip is small enough to be neglected, we can relate the linear and angular velocity of each wheel using the following relationship [9]:

$$v_{ix} = r_i w_i \tag{2.14}$$

here r_i is the radius of the wheel.

When the vehicle turns, at each time instant an Instantaneous Center of Rotation (ICR) is generated [2] as seen in Figure 2.4. We can define the vector from the ICR to the center C as $\vec{r_c} = [r_{cx}\ r_{cy}]^T$. Similarly, we can define a vector from the ICR to the center of each contact point P_i as : $\vec{r_{ic}} = [r_{icx}\ r_{icy}]^T$. The angular velocity of the vehicle can be calculated knowing that $\vec{v} = \vec{w} \times \vec{r_c}$. The wheels will have the same angular rotation around the ICR, so by using trivial algebra, we get to the following relationship:

$$\frac{v_x}{-r_{cy}} = \frac{v_{ix}}{-r_{icy}} = \frac{v_y}{r_{cx}} = \frac{v_{iy}}{r_{icx}} = \dot{\psi} \tag{2.15}$$

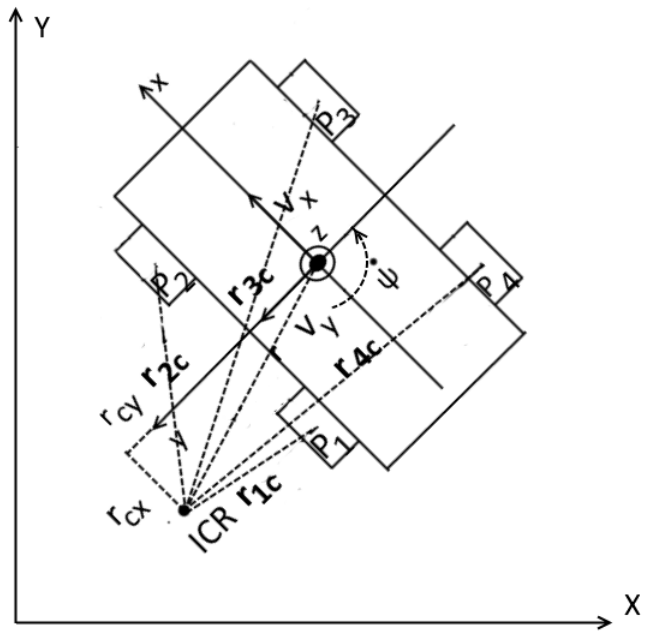


Figure 2.4: When the vehicle turns an ICR appears at each time instant. Here we can see its coordinates in the local frame.

The coordinates of the ICR in the local reference frame L are:

$$\vec{ICR} = [x_{ICR}\ y_{ICR}]^T = [-r_{cx}\ \ -r_{cy}]^T \tag{2.16}$$

which allows us to rewrite Equation (2.15) as follows:

$$\frac{v_x}{y_{ICR}} = \frac{v_y}{-x_{ICR}} = \dot{\psi} \tag{2.17}$$

The lateral velocity of the COM can be obtained using equation:

$$v_y = -x_{ICR}\dot{\psi} \tag{2.18}$$

Now, we need to relate the velocity of the COM with the velocity of each wheel. To that purpose, from the geometry seen in Figure 2.4, the following relationships hold:

$$\begin{aligned} r_{1cy} &= r_{2cy} = r_{cy} + w/2; \\ r_{3cy} &= r_{4cy} = r_{cy} - w/2; \\ r_{2cx} &= r_{3cx} = r_{cx} + a; \\ r_{1cx} &= r_{4cx} = r_{cx} - b; \end{aligned} \tag{2.19}$$

By combining Equations (2.19) and (2.15), we obtain the following equalities:

$$\begin{aligned} v_{1x} &= v_{2x} \\ v_{3x} &= v_{4x} \\ v_{2y} &= v_{3y} \\ v_{1y} &= v_{4y} \end{aligned} \tag{2.20}$$

For control purposes, we want to use the input vector $u = [v_x \quad \dot{\psi}]^T$. Using Equations (2.20) with (2.15) we can obtain a relationship between the wheels' velocities and u as follows:

$$\begin{bmatrix} v_{1x} \\ v_{2x} \\ v_{3x} \\ v_{4x} \end{bmatrix} = \begin{bmatrix} 1 & -w/2 \\ 1 & -w/2 \\ 1 & w/2 \\ 1 & w/2 \end{bmatrix} \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} \tag{2.21}$$

$$\begin{bmatrix} v_{1y} \\ v_{2y} \\ v_{3y} \\ v_{4y} \end{bmatrix} = \begin{bmatrix} 0 & -x_{ICR} - b \\ 0 & -x_{ICR} + a \\ 0 & -x_{ICR} + a \\ 0 & -x_{ICR} - b \end{bmatrix} \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} \tag{2.22}$$

From Equations (2.21) and (2.14) it is also trivial to obtain the wheels' angular velocities as:

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 1 & -\frac{w}{2} \\ 1 & -\frac{w}{2} \\ 1 & \frac{w}{2} \\ 1 & \frac{w}{2} \end{bmatrix} \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} \tag{2.23}$$

Therefore, The vector u can be interpreted as the reference velocities from which the reference wheels' angular speeds can be obtained to act as the inputs for each motor's control system.

Non-holomonic Constraint

It is important to study the effect of kinematic constraints that limit the movements of the vehicle, although it is possible to reach any desirable configuration by planning an appropriate set of manoeuvres. This constraints can be of two types: holomonic and non-holomonic.

The vector of coordinates $q = [X\ Y\ \psi]^T$ describes the configuration of the vehicle at any instant of time. We can say that a constraint is holomonic [16] if it can be described in the form:

$$h_i(q) = 0 \tag{2.24}$$

A kinematic constraint $a_i(q, \dot{q}) = 0$ depends on the vector of coordinates and their derivatives. Under the assumption that the kinematic constraints are linearly dependant on their velocities, the restrictions are usually expressed in the Pfaffian form as follows:

$$a_i^T(q)\dot{q} = 0 \tag{2.25}$$

A set of restrictions in the form (2.24) implies that we will have an equal number of kinematic constraints:

$$\frac{\partial h_i(q)}{\partial t} = \frac{\partial h_i(q)}{\partial q}\dot{q} = 0 \tag{2.26}$$

However, starting from a set of kinematic constraints ,they may or may not be integrable i.e., come back to the form (2.24). When it is not integrable in at least one constraint, we say that the mechanical system is non-holomonic.

In [3] they establish that, for a skid-steering vehicle, x_{ICR} must remain between the car's wheelbases. If it goes out, the vehicle would skid along the y-axis, losing control. Therefore, the following non-holomonic constraint must be introduced:

$$v_y + x_{ICR}\dot{\psi} = 0 \tag{2.27}$$

which, using (2.8), can be written in the Pfaffan form as follows:

$$[-sin\psi \quad cos\psi \quad x_{ICR}][\dot{X} \quad \dot{Y} \quad \dot{\psi}]^T = A(q)\dot{q} = 0 \tag{2.28}$$

Since $\dot{q}$ always belongs to the null space of $A(q)$ [16] we can rewrite $\dot{q}$ in order to depend on our control input vector u using Equations (2.27) and (2.8) :

$$\dot{q} = \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -x_{ICR} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} = S(q)u(t) \tag{2.29}$$

$$S(q) = \begin{bmatrix} \cos(\psi) & x_{ICR}\sin(\psi) \\ \sin(\psi) & -x_{ICR}\cos(\psi) \\ 0 & 1 \end{bmatrix} \qquad u(t) = \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} \tag{2.30}$$

where:

$$A(q)S(q) = S(q)^T A(q)^T = 0; \tag{2.31}$$

The admissible trajectories for the vehicle must be the ones that satisfies the non-linear system (2.29).

It is also important to point out that $S(q)$ relates the derivative of the vector of coordinates $\dot{q}$ with the control input vector u, therefore, it can be used as a transformation matrix from the control inputs to the velocities in G, while $S(q)^{-1}$ will do the inverse transformation.

Lateral Velocity Estimation

It is interesting to note how the value of x_{ICR} changes drastically the kinematics. However, it does not take arbitrary values. In fact, according to [3], the value must be within the vehicle's axles and allows us to approximate the lateral velocity of the vehicle using Equation (2.18).

The exact value of the ICR is unknown, and although its instant velocity is zero, the position may change over time, varying v_y. In [9], the authors proposed to substitute the unknown value x_{ICR} by a constant value x_o selected by the user, bounded between the vehicle's dimensions $(a, -b)$, that is not related to any geometrical configuration of the system:

$$v_y = -x_o\dot{\psi} \tag{2.32}$$

Several authors have used other approaches. In particular, [19] during their experiments have found a non-linear relationship between the turning radius and the difference between the wheel velocities. In order to find a relationship, they introduced a non-dimensional value γ as follows:

$$\gamma = \frac{\omega_l + \omega_r}{\omega_l - \omega r} \tag{2.33}$$

and then, the turning radius R becomes $R(\gamma) = K\gamma$ where K is a constant value estimated from experimental results. We can prove that this solution is related to the one propossed by [3]. First, under the main hypothesis that the wheels on the same side rotate with the same angular velocity [19], the vector u is related to the sides' angular velocities using Equation (2.14) and Equation (2.21):

$$\begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} = r \begin{bmatrix} \frac{w_l+w_r}{2} \\ \frac{-w_l+w_r}{w} \end{bmatrix} \tag{2.34}$$

where w_l and w_r are the angular velocities of the left and right side of the car respectively.

We can rewrite γ using Equation (2.34) to obtain:

$$2v_x = \gamma w \dot{\psi} \tag{2.35}$$

Recalling from (2.15), v_x can be written as $-y_{ICR}v_y/x_{ICR}$.Therefore γ is related to x_o as follows:

$$v_y = -\gamma w \frac{x_{ICR}}{2y_{ICR}} \dot{\psi} = -x_o \dot{\psi} \tag{2.36}$$

which demonstrates the fact that both solutions are similar between each other and that the value of x_o is a parameter that has to be obtained experimentally. Moreover, they have found that acceleration or deceleration of the vehicle changes the behaviour of the turning making it understeer or oversteer.

A more complex method followed by [20] is to calculate the value of x_{ICR} using and empirical approach with an IMU and an Extended Kalman Filter, taking the measurements in real time, which gives better precisions than other methods with the drawback of the need of another sensor. In their results they have found that the value of x_{ICR} does not change drastically with the surface condition, oscillating around a nominal value, being able to do a fitting curve of $x_{ICR}(\gamma)$.

On several tests, researchers from [17] have found measurements of x_{ICR} that go outside of the fitting curve in the form of spikes, which corresponds to sliding's situations in the vehicle, concluding that the increase of this value from its nominal value is an indicator of such behaviour.

2.3 Dynamics

2.3.1 DC Brushless Dynamic Model

Introduction

The DC brushless motor can be defined as an AC machine in which, at steady state, the rotation of the shaft is synchronized with the frequency of the supply current. It is fed by a DC source by means of a square wave inverter, which produces the AC electric signal to power the motor. The shape of the electric current depends on the realization of the windings, being the sinusoidal and trapezoidal the most common waveforms. Besides, the current keeps the characteristic of being an alternating signal.

The DC brushless motor contains permanent magnets integrated in the steel rotor in order to create a constant magnetic field. The stator carries windings connected to the inverter to produce a rotating magnetic field. As it rotates around a fixed armature, problems related to wiring of the moving armature are removed. Similarly, as the armature is isolated in its housing, all issues related to wearing and cooling are also reduced. These advantages have as counterpart the increasing complexity and more expensive control techniques.

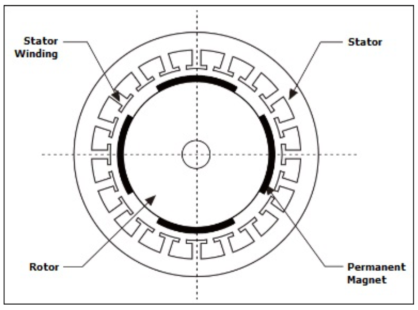


Figure 2.5: PMSM internal structure with two polar couples.

The brushless motors are commonly used in high precision applications in which an accurate speed is required, such as in synchronous clocks or positioning systems. One of their principal features is the high efficiency which combined with their high precision, make them a special type of motors. In addition, it is a machine capable of running at constant speed regardless the load acting on it, ie as long as the torque is within the range of the nominal torque, the motor will be able to move at the desired speed.

Dynamic Model

First lets do the analysis for the first winding, for a brushless DC motor suppose that the time interval in which the first electromotive force e_{s1} is constant is 1/3 of the period T. As the idea is to simplify the analysis, we define the angle equal to 120 degrees. Now, the winding is supplied by a square wave current (with a value of Id, which is constant during 120 degrees) and the total mechanical power is constant. Figure 2.6 shows the trend of e_{s1}.

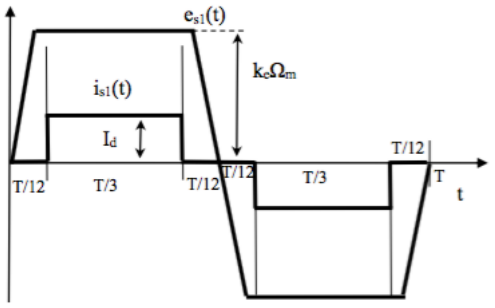


Figure 2.6: Electromotive force and phase current waveform.

At each time instant, two currents are different from zero, so the mechanical power is:

$$P_m = 2k_e\omega I_d \tag{2.37}$$

where k_e is the voltage constant of the motor and ω is the mechanical speed of the rotor. The electromagnetic torque is given by:

$$\tau_e = 2k_e I_d \tag{2.38}$$

The simplified equation of the mechanical energy balance is:

$$\dot{\omega} = \frac{n_p}{J}(\tau_e - \tau_m) \tag{2.39}$$

where $\dot{\omega}$ is the angular acceleration, n_p is the number of pole pairs, J represents the rotor inertia, τ_e the electric torque and τ_m the mechanical torque.

Figure 2.7 shows that for a period T, we can divide the voltage signal wave of the DC brushless motor into six equally spaced sectors, each of them with T_{sect} duration. In the graphic, in each sector only two windings are powered. For instance, in sector 6 the current flows only in s3 as input and in s2 as output.

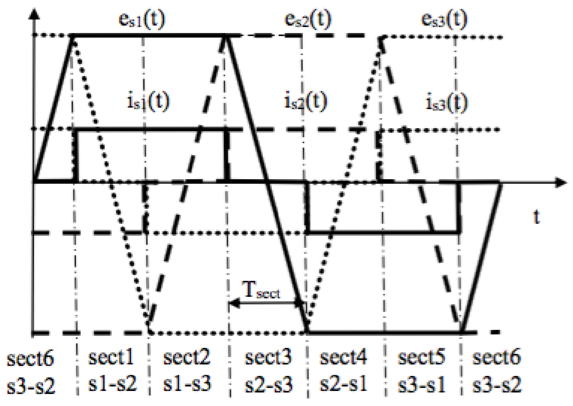


Figure 2.7: Sectors depending on the windings fed.

The equivalent circuit in the terminals of the power converter is the same for all sectors, then we only need to switch the sectors seen by the converter. This is done by measuring the rotor position in order to feed the right windings. In Figure 2.8 we can see the equivalent electrical circuit for each sector.

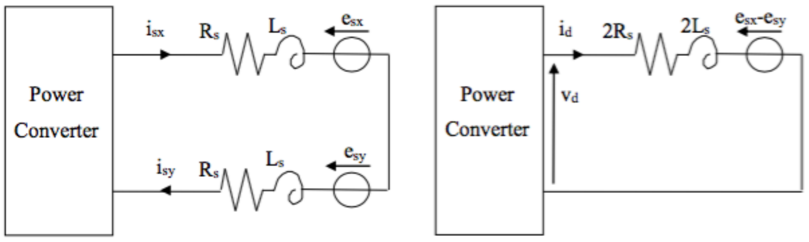


Figure 2.8: Equivalent DC brushless motor circuit.

From Figure 2.8, the difference $e_{sx} - e_{sy}$ is constant and is described by two back emf in series:

$$e_{sx} - e_{sy} = 2k_e\omega \tag{2.40}$$

Thus, the idea is to control the current i_d: it has to be controlled as a constant value I_d. The dynamic equation for a DC brushless motor is:

$$v_d = 2R_s i_d + 2L_s \dot{i}_d + 2k_e\omega = 2R_s i_d + 2L_s \dot{i}_d + E \tag{2.41}$$

By analysing Equation (2.41), it can be seen that the condition in order to have constant torque, i.e., constant value I_d, is that the line-to-line emf has to be constant within the sector.

Final Modeling of the Motor

Under the assumptions that at every time instant, two windings are always fed and neglecting the switching effect of the inverter and its non-linearities, we can propose an equivalent machine that, although is not real, retains the relevant dynamics for out modelling.

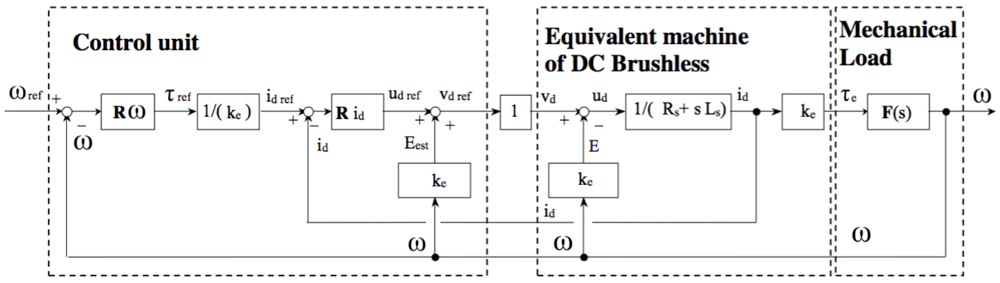


Figure 2.9: DC brushless motor model including the cascade control loops.

The whole model of the system is shown in Figure 2.9. In this model, k_e, L_s and R_s are equal to twice the values for a single winding. Considering that the wheel is directly coupled to the rotor, J_m is the whole inertia and B is the friction coefficient of the axle, which can be obtained using the mechanical time constant $T = L/R$ of the motor, the expression is given by: $B = J_m/T$. $F(s)$ denotes the mechanical transfer function attached to the wheel and appears on Equation (2.42).

$$F(s)\tau_e = \omega = \frac{\tau_e - \tau_m}{J_m s + B} \tag{2.42}$$

We need to obtain the mechanical torque τ_m from Equation (2.42), which will feed the vehicle dynamics block. To do that, we can rearrange the expresion as follows:

$$\tau_m = \tau_e - \omega(J_m s + B) \tag{2.43}$$

The right side of Equation (2.43) has two terms: the first one is the electromagnetic torque, while the second part is the effect due to the dynamics of the wheel. The difference between both will be the amount of torque fed by each motor to the vehicle. It is important to note that this equation is dependant on the wheel angular velocity and its derivative, which, can be obtained for each wheel from the vehicle's COM velocities and relationship (2.23).

2.3.2 Vehicle Dynamics

Gravitational Force on the Vehicle

When the vehicle is not moving on a planar surface, the gravity affects the forces applied on the car. The easiest way to model this effect is to find the projection of the gravity vector $\vec{g_G}$ on the frame L.

On Section 2.2.1 we obtained the rotation matrix $R_{L-G}(\alpha, \beta, \rho)$ which projects a vector from G onto L. On the Global frame, the gravity vector always points on the Z direction. We can define the vector $\vec{g_G} = [0 \quad 0 \quad g]^T$ and its projection onto L as $\vec{g_L} = [g_x \quad g_y \quad g_z]^T$. The Cardan angles will be denoted as follow:

1. α, is rotation of the surface in which the vehicle runs in the X axis.
2. β, is the rotation of the surface in which the vehicle runs in the Y axis.
3. $\rho =$ is the vehicle's heading, therefore is exactly ψ.

Then we can denote $\vec{g_L}$ as:

$$\vec{g_L} = R_{L-G} = R_\psi R_\beta R_\alpha \vec{g_G} \tag{2.44}$$

Finally, we can define the gravity's resistive matrix as:

$$R_g = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = \begin{bmatrix} mg_x \\ mg_y \\ mg_z \end{bmatrix} = \begin{bmatrix} F_{gx} \\ F_{gy} \\ F_{gz} \end{bmatrix} \tag{2.45}$$

Forces on the Wheels

In each wheel the forces acting are seen on Figure 2.10:

1. F_i, the active force related to the torque applied by the motor i $Fi = \frac{\tau_i}{r}$;
2. N_i, the normal force applied on wheel i(going out of the figure);
3. F_{li}, The resistive force applied lateral to the wheel plane;
4. F_{si}, the resistive force applied longitudinal to the wheel plane.

The resistive forces are the result of the interaction between the ground and the wheel. In particular, F_{si} will be denoted as the rolling resistance force. When the tire rolls on a surface, the contact area undergoes a deflection,

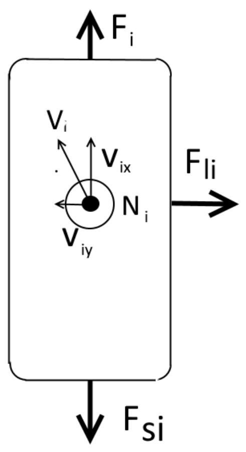


Figure 2.10: Forces acting on the wheel. The normal force N_i is pointing outside of the figure.

due to the fact that the tire is not perfectly elastic, part of the energy spent in the deformation is not restored, this energy loss is translated into F_{si} [6]. It can be written as:

$$F_{si}(v_{ix}) = \mu_s N_i sgn(v_{ix}) \tag{2.46}$$

Where μ_s denotes the rolling resistance coefficient and $sgn(v_{ix})$ simply defines the sign of the velocity.

The lateral resistance force is directly proportional to the Normal load in the wheel by the term μ_l. Therefore we can write is as:

$$F_{li}(v_{iy}) = \mu_l N_i sgn(v_{iy}) \tag{2.47}$$

The normal forces depend on the mass, the position of the COM and the projection of the gravity vector onto z. We can approximate the normal force on each axle as follows:

$$N_{back} = \frac{a}{(a+b)} mg_z \quad N_{front} = \frac{b}{(a+b)} mg_z \tag{2.48}$$

Using the assumption that this forces are evenly distributed between its wheels, the normal force on each wheel can be approximated as:

$$N_{1,4} = \frac{a}{2(a+b)} mg_z \quad N_{2,3} = \frac{b}{2(a+b)} mg_z \tag{2.49}$$

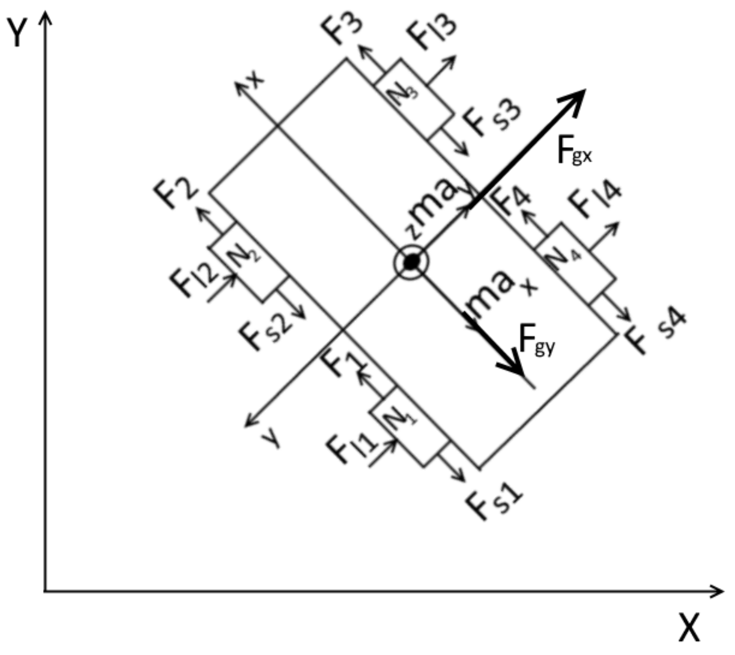


Figure 2.11: Free body diagram of the vehicle in which main forces are shown.

Equation of Motion

The energy approach is going to be used to obtain the equation of motion of the vehicle. First, lets write the kinetic energy as:

$$T = \frac{1}{2}m\vec{v}^T\vec{v} + \frac{1}{2}J\vec{w}^T\vec{w} \tag{2.50}$$

where m is the mass of the vehicle and J the inertial term around the COM. Considering that $\vec{v}^T\vec{v} = v_x^2 + v_y^2 = \dot{X}^2 + \dot{Y}^2$ and $\vec{w}^T\vec{w} = \dot{\psi}$, we can write the equation in terms of the derivative of the vector coordinate $\dot{q} = [\dot{X} \quad \dot{Y} \quad \dot{\psi}]^T$:

$$T = \frac{1}{2}m(\dot{X}^2 + \dot{Y}^2) + \frac{1}{2}J\dot{\psi}^2 = \frac{1}{2}\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix}\begin{bmatrix} \dot{X}^2 \\ \dot{Y}^2 \\ \dot{\psi}^2 \end{bmatrix} \tag{2.51}$$

Now, deriving T with respect to $\dot{q}$ and then with respect to time, we can obtain the inertial term of the equation of motion.

$$\frac{d}{dt}(\frac{\partial T}{\partial \dot{q}}) = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix}\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{\psi} \end{bmatrix} = [m]\ddot{q} \tag{2.52}$$

The active and reactive forces acting on the vehicle are the forces delivered by each one of the motors and the resistive forces on each wheel, as shown in Figure 2.11. They can be expressed as:

$$\begin{cases} \sum F_x = \sum\limits_{i=1}^{4} F_{ix} - \sum\limits_{i=1}^{4} F_{si} - F_{gx} \\ \sum F_y = -\sum\limits_{i=1}^{4} F_{li} - F_{gy} \\ \sum M_{COM} = \frac{w}{2}(\sum\limits_{i=3}^{4} F_i - \sum\limits_{i=1}^{2} F_i) - a\sum\limits_{i=2}^{3} F_{li} + b\sum\limits_{i=1,4} F_{li} + \\ +\frac{w}{2}(\sum\limits_{i=1}^{2} F_{si} - \sum\limits_{i=3}^{4} F_{si}) \end{cases} \tag{2.53}$$

Using the rotation matrix (2.9), we can express the forces with respect to vector q and its derivatives. It will also be defined a resistive force matrix $[R]$ and an active force matrix $[F]$ as:

$$[R] = \begin{bmatrix} F_{rX} \\ F_{rY} \\ M_{r\psi} \end{bmatrix} = \begin{bmatrix} (\sum\limits_{i=1}^{4} F_{si} + F_{gx})\cos\psi - (\sum\limits_{i=1}^{4} F_{li} + F_{gy})\sin\psi \\ (\sum\limits_{i=1}^{4} F_{si} + F_{gx})\sin\psi + (\sum\limits_{i=1}^{4} F_{li} + F_{gy})\cos\psi \\ a\sum\limits_{i=2}^{3} F_{li} - b\sum\limits_{i=1,4} F_{li} + \frac{w}{2}(\sum\limits_{i=3}^{4} F_{si} - \sum\limits_{i=1}^{2} F_{si}) \end{bmatrix} \tag{2.54}$$

$$[F] = \begin{bmatrix} \sum\limits_{i=1}^{4} F_i \cos(\psi) \\ \sum\limits_{i=1}^{4} F_i \sin(\psi) \\ \frac{w}{2}(-F_1 - F_2 + F_3 + F_4) \end{bmatrix} \tag{2.55}$$

such that the equation of motion of the system becomes:

$$[m]\ddot{q} + [R] = [F] \tag{2.56}$$

However, it is necessary to rewrite (2.56) in terms of the motor torques and control input vector. By recalling that $F_i = \frac{\tau_i}{r_i}$ and assuming that the radius of all the wheels is the same, τ_l and τ_r are the sum of the torques of the left and right wheels, respectively. Therefore, $[F]$ becomes:

$$\begin{cases} [F] = [B]\tau \\ [B] = \frac{1}{r}\begin{bmatrix} \cos(\psi) & \cos(\psi) \\ \sin(\psi) & \sin(\psi) \\ -\frac{w}{2} & \frac{w}{2} \end{bmatrix} \\ \tau = \begin{bmatrix} \tau_l \\ \tau_r \end{bmatrix} = \begin{bmatrix} \tau_1 + \tau_2 \\ \tau_3 + \tau_4 \end{bmatrix} \end{cases} \tag{2.57}$$

where $[B]$ is the input transformation matrix. Now, with (2.57) the equation of motion (2.56) will be:

$$[m]\ddot{q} + [R] = [B]\tau \tag{2.58}$$

Which describes the equation of motion of a free body and does not includes the non-holomonic constrains. In order to do that, a vector of Lagranian multipliers λ is introduced [3]:

$$[m]\ddot{q} + [R] = [B]\tau + A(q)^T\lambda \tag{2.59}$$

where $A(q)$ was defined on Equation (2.28). We can use relationship (2.29) to eliminate λ. Multiplying all by $S(q)^T$ we get:

$$S(q)^T[m]\ddot{q} + S(q)^T[R] = S(q)^T[B]\tau + S(q)^TA(q)^T\lambda \tag{2.60}$$

and recalling (2.31): $S(q)^TA(q)^T\lambda$ is zero.

We now express the acceleration vector $\ddot{q}$ in terms of u and $\dot{u}$ as follows:

$$\ddot{q} = \dot{S}(q,\dot{q})u + S(q)\dot{u} \tag{2.61}$$

By introducing Equation (2.61) into (2.60) and rearranging the terms we obtain the equation of motion in terms of u and $\dot{u}$:

$$\bar{M}\dot{u} + \bar{D}u + \bar{R} = \bar{B}\tau \tag{2.62}$$

where:

$$\bar{D} = S(q)^t[m]\dot{S}(q,\dot{q}) = mx_{ICR}\begin{bmatrix} 0 & \dot{\psi} \\ -\dot{\psi} & \dot{x}_{ICR} = 0 \end{bmatrix} \tag{2.63}$$

$$\bar{M} = Sq^T[m]Sq = \begin{bmatrix} m & 0 \\ 0 & mx_{ICR}^2 + J \end{bmatrix} \tag{2.64}$$

$$\bar{B} = Sq^TB = \frac{1}{r}\begin{bmatrix} 1 & 1 \\ -\frac{w}{2} & \frac{w}{2} \end{bmatrix} \tag{2.65}$$

$$\bar{R} = Sq^TR = \begin{bmatrix} F_{rX} \\ x_{ICR}F_{rY} + M_{r\psi} \end{bmatrix} \tag{2.66}$$

2.3.3 Final Model Block Diagram

Finally, all the kinematic and dynamic relationships were presented, now we can use the diagram on Figure 2.12 to present the model of a skid-steering robot. Each block is identified by a letter and they represent the following:

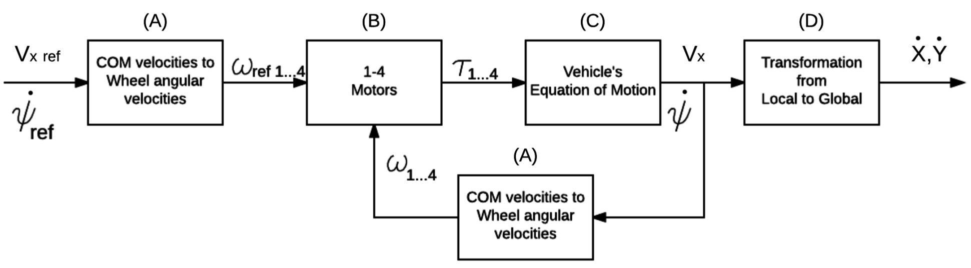


Figure 2.12: Final block diagram with the model of the vehicle

1. (A) COM's velocities to wheel angular velocities: implement the Equation (2.23).
2. (B) 1...4 motors: implement the motor dynamic model presented in Figure 4.1 and Equation (2.43) four times, one per each motor in order to generate the output torques.
3. (C) Vehicle's equation of motion: it contains the dynamics of the vehicle given by the Equation (2.62). It takes as inputs the torques of the motors and as outputs generates the velocity and acceleration of the COM.
4. (D) Transformation fromlocal to global: uses the transformation matrix $S(q)$ presented in Equation (2.29) to convert the velocities in the reference frame L to G.

The input vector $u_{ref} = [v_{xref} \quad \dot{\psi}_{ref}]^T$ acts as inputs, which are transformed by (A) into the vector $\omega_{ref1...4}$ that represents the wheels' angular velocities references which with the block (B) will become into the torques to drive the vehicle. They will feed the block (C) which is the equation of motion of the vehicle to output the real $u_{real} = [v_x \quad \dot{\psi}]^T$ and finally in (D) they are transformed into the velocities in the global reference frame.

2.4 Trajectory Generator

On Section 2.2.2, a non-holomonic constraint that limits the movements of the vehicle was introduced. We also considered that $\dot{q} = [\dot{X} \quad \dot{Y} \quad \dot{\psi}]^T$ are the velocity vectors of the system that will always belong to the null space of $A(q)$ and that vector can be rewritten with respect to the local commands $u(t) = [v_x(t) \quad \psi(t)]^T$ as:

$$\dot{q} = \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & x_{ICR}\sin(\psi) \\ \sin(\psi) & -x_{ICR}\cos(\psi) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ \dot{\psi} \end{bmatrix} = S(q)u(t) \tag{2.67}$$

Assume now that we want to plan a trajectory that leads the vehicle from an initial configuration $q_(t_i) = [X_i \quad Y_i \quad \psi_i]^T$ to a final configuration $q(t_f) = [X_f \quad Y_f \quad \psi_f]^T$ in the time interval (t_i, t_f). We can split the desired trajectory in two parts: a geometrical path $q(s)$ such that $\frac{dq(s)}{ds} \neq 0 \quad \forall s$ and a timing law $s = s(t)$. We can then write $\dot{q}$ as:

$$\dot{q} = \frac{dq}{ds}\frac{ds}{dt} = q'\dot{s} \tag{2.68}$$

Then the admissible trajectories are the solution to the non-linear system:

$$q' = S(q)u' \tag{2.69}$$

where $u' = [v_x' \quad \omega']^T$ is the vector of geometric inputs such that $u(t) = u'\dot{s}$.

In our particular case, the admissible path for a skid steering system are the solution of the system of equation:

$$\begin{cases} X' = v_x'\cos(\psi) + \omega' x_{ICR}\sin(\psi) \\ Y' = v_x'\sin(\psi) - \omega' x_{ICR}\cos(\psi) \\ \psi' = \omega' \end{cases} \tag{2.70}$$

From which we need to obtain the geometrical inputs. To do that, we can rewrite 2.70 as follows:

$$\begin{cases} X' - \omega' x_{ICR}\sin(\psi) = v_x'\cos(\psi) \\ Y' + \omega' x_{ICR}\cos(\psi = v_x'\sin(\psi)) \\ \psi' = \omega' \end{cases} \tag{2.71}$$

Power square of the first equation and after add the second, we obtain the following:

$${v_x'}^2 = X'^2 + Y'^2 + \omega'^2 x_{ICR}^2 + 2\omega' x_{ICR}(Y'\cos(\psi) - X'\sin(\psi)) \tag{2.72}$$

Using Equation (2.8) and (2.27) we can rewrite v_y' as:

$$v_y' = Y'\cos(\psi) - X'\sin(\psi) = -x_{ICR}\omega' \tag{2.73}$$

Introducting 2.73 into 2.72 we obtain:

$$v_x'^2 = X'^2 + Y'^2 + \omega'^2 x_{ICR}^2 + 2\omega' x_{ICR}(-x_{ICR}\omega') = X'^2 + Y'^2 - \omega'^2 x_{ICR}^2 \tag{2.74}$$

which means that the admissible geometric input v_x' must be:

$$v_x' = \pm\sqrt{X'^2 + Y'^2 - \omega'^2 x_{ICR}^2} \tag{2.75}$$

in which the sign of v_x' depends if the vehicle is going forward or backwards.

With respect to ω' we first write ψ as $\psi(s) = Atan2(Y', X') + k\pi$ where k = 0.1 depends if the path is followed forward or backward. Now, deriving with respect to s becomes:

$$\frac{d\psi(s)}{ds} = \omega' = \frac{dAtan2(Y', X')}{ds} = \frac{-Y'}{X'^2 + Y'^2}\frac{dX'}{ds} + \frac{X'}{X'^2 + Y'^2}\frac{dY'}{ds} \tag{2.76}$$

which allows us to get to the final relationship:

$$\omega' = \frac{X'Y'' - Y'X''}{X'^2 + Y'^2} \tag{2.77}$$

Equations (2.75) and (2.77) allows us to obtain the geometrical inputs for a given desired path. Now, all that is necessary is to denote a given timing law of the system.

2.4.1 Experimental Desired Path

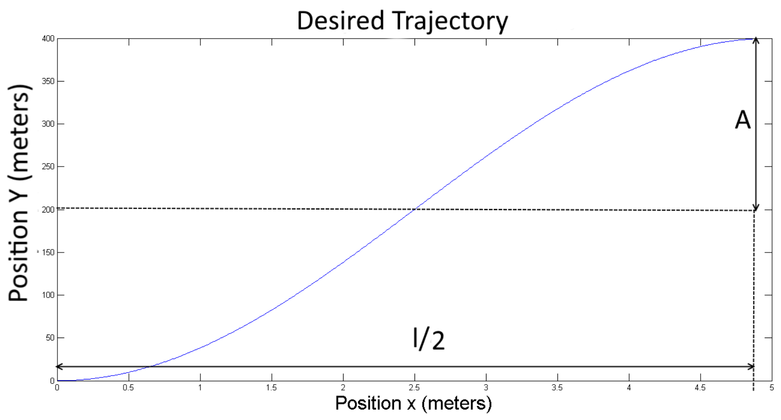


Figure 2.13: Desired trajectory for the experiments.

For our experiments we would like a trajectory like the one shown in Figure 2.13. In which we simulate the path of a vineyard starting from a initial configuration parallel to the x axis at the bottom of a line, then passing through the hallway between bushes and ending in the top part of the line with the same orientation as the beginning. We can approximate such trajectory with an equation of the form:

$$Y(X) = -A\cos(\frac{2\pi}{l}X) + A \tag{2.78}$$

in which $'2A'$ denotes the total length of the corridor and $\frac{l}{2}$ is its width.

The total linear distance S_t that the vehicle must run is the arc length of the trajectory, that is:

$$S_t = \int_0^{l/2} \sqrt{1 + \frac{dY}{dX}^2} dx = \int_0^{l/2} \sqrt{1 + (\frac{2\pi}{l})A\sin(\frac{2\pi}{l}x)^2} dx \tag{2.79}$$

However, in this implementation $Y = Y(X)$, meaning that we need to find a way to relate Y with the linear distance $S(t)$. To this aim, we can decompose it into a its contribution in the Y and X axis. In other words:

$$S(t) = \sqrt{S_X(t)^2 + S_Y(t)^2} \tag{2.80}$$

Lets propose a linear relationship between $X(t)$ and $S(t)$. At the end of the trajectory at time t_f, in the X-axis it must have gone in the from 0 to $l/2$ so it can be said that:

$$S_X(t_f) = C_X S_t = l/2 \tag{2.81}$$

from which we can obtain the constant C_X and then we can write X as:

$$X(t) = X(S) = C_X S(t) \tag{2.82}$$

Now, both Y and X are related to the linear trajectory of the vehicle.

It is necessary to find the geometrical derivatives of Y and X. The derivatives with respect to t can be written as follows:

$$\begin{cases} \dot{Y}(t) = \frac{dY}{dt} = \frac{dY}{dX}\frac{dX}{dS}\frac{dS}{dt} = Y'\dot{S}(t) \\ \dot{X}(t) = \frac{dX}{dt} = \frac{dX}{dS}\frac{dS}{dt} = X'\dot{S}(t) \\ \ddot{Y}(t) = Y''\dot{S}(t) \\ \ddot{X}(t) = X''\dot{S}(t) \end{cases} \tag{2.83}$$

Applying Equation (2.83) on (2.78) and (2.82), and after rearranging the system we obtain the following relationships:

$$\begin{cases} X' = C_X \\ X'' = 0 \\ Y' = \frac{2\pi}{l} C_x A \sin(\frac{2\pi}{l} X(S)) \\ Y'' = (\frac{2\pi}{l} C_x)^2 A \cos(\frac{2\pi}{l} X(S)) \end{cases} \tag{2.84}$$

With Equation (2.84) applied on (2.75) and (2.77), we can obtain the necessary geometrical inputs in order to obtain the desired movement of the vehicle.

2.4.2 Timing Law

Once the geometrical trajectory is set, it is necessary to obtain the timing law that the vehicle must follow, i.e. the equation that fulfills the set of desired constraints. Proposing a cubic timing law of the form:

$$S(t) = At^3 + Bt^2 + Ct + D \tag{2.85}$$

its first derivative will become a parabolic velocity profile and the second derivative a linear acceleration profile as follows:

$$\begin{cases} \dot{S}(t) = 3At^2 + 2Bt + C \\ \ddot{S}(t) = 6At + 2B \end{cases} \tag{2.86}$$

The constants 'A', 'B', 'C' and 'D' will define the selected velocities and acceleration subjected to the following constraints:

$$\begin{cases} S(0) = S_0 \quad \text{The initial position of the vehicle} \\ S(Tf) = S_t \quad \text{The final position of the vehicle at time } Tf \\ \dot{S}(0) = V_0 \quad \text{Initial Velocity} \\ \dot{S}(Tf) = V_f \quad \text{The final desired Velocity} \end{cases} \tag{2.87}$$

With this constraints set by the user and the Equations (2.85) and (2.86) it is possible to build a system of equations that solves the constants for the velocity, acceleration and position profiles.

For example, consider a vineyard with 400 meters length and 10 meters space in between with the trajectory shown in Figure 2.14. With the trajectory conditions shown in Table 2.1.

The timing law, the velocity and acceleration profile calculated are shown in Figure 2.15 in which, as expected, a parabolic velocity profile and a linear acceleration profile appears, with initial distance equal to zero and final

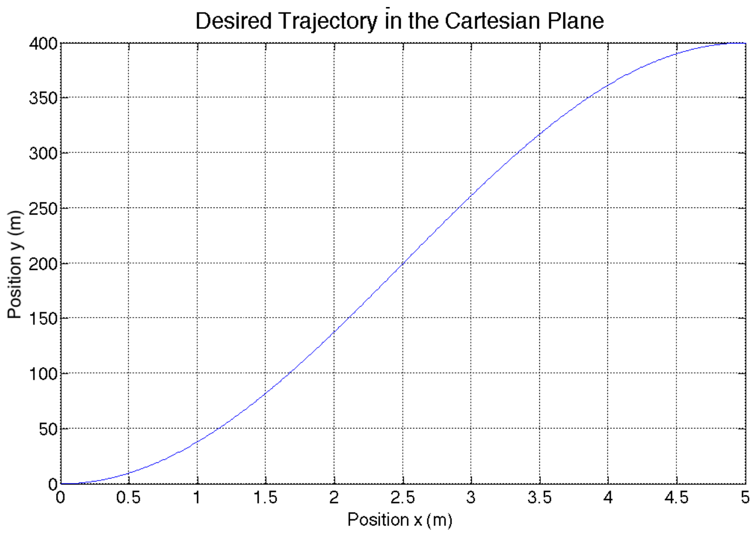


Figure 2.14: Desired path used as example.

A (meters)	200
l (meters)	10
Initial Position S_0 (meters)	0
Final Position S_t (meters)	400.09 (from Equation (2.79))
Initial Velocity V_0 (meters/sec)	0
Final Velocity V_f (meters/sec)	0
total time Tf (seconds)	80

Table 2.1: Desired conditions for the path used to calculate the timing law and the input commands to the vehicle.

distance equal to S_t. From this timing law, the input commands to the vehicle are calculated, it is shown in Figure 2.16. We can see that the initial and final velocities are zero, with a peak in the middle of the trajectory. We can also observe that the vehicle is initial pointing parallel to the X axis so at the beginning a peak of angular velocity appears in order to align himself with the trajectory. The same behavior appears at the end of the trajectory in order to end again with the same orientation as the beginning.

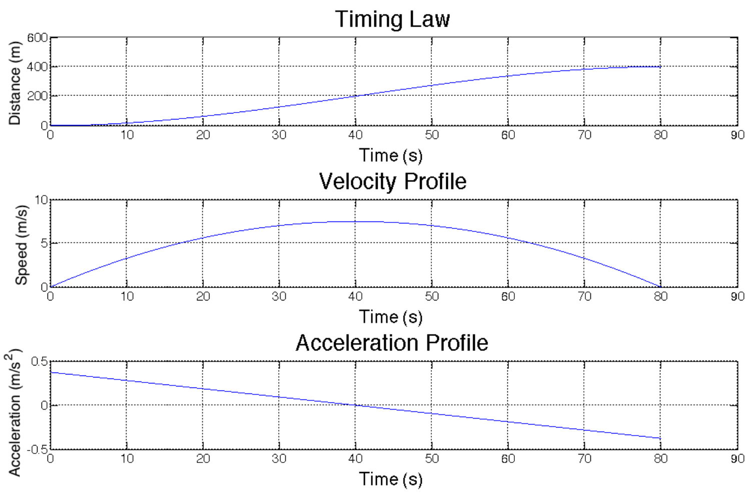


Figure 2.15: Top: Desired timing law, Middle: Velocity profile,Bottom: Acceleration profile.

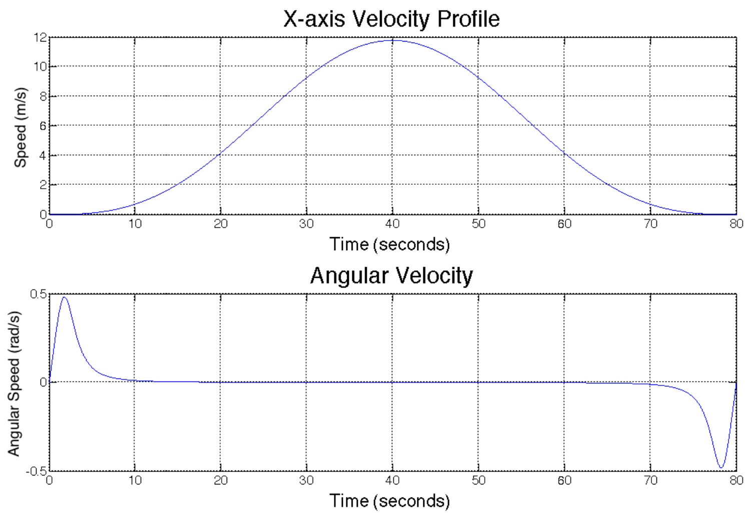


Figure 2.16: The input commands to give to the vehicle in order to obtain the desired trajectory. Top: Velocity in the linear direction, Bottom: angular velocity of the system.

Chapter 3

Design and Selection of Controllers

The design and selection of controllers plays an essential role because the behavior of the vehicle depends mostly on them. Different control methodologies could be implemented but most of them are too complex for the system or are not suitable for the performance required.

There is a general agreement that the most effective control scheme for electrical drives is a cascaded or nested structure with a fast inner control loop, limiting the torque, to which an outer speed control loop is superimposed [10], this multi-loop has proved to be very flexible. For instance, more inner or outer control loops can be added if there is a need for control of other variables like position or acceleration.

A cascade control scheme can work only if the bandwidth of the inner loops are much faster than the outer loops. This is necessary because an outer loop can only perform well if the inner loops execute quickly the required commands. This methodology has the advantage of dealing with only a small part of the system instead of the whole, making it easier to tune. Finally, for load disturbances entering the system, cascade control has proven to be superior to other control methodologies because the next controller will detect the disturbance and counteract it [10].

In the general case in which the time constant of the electrical transfer function is smaller than the mechanical counterpart, the inner loop will be the current control and the outer will be the speed control.

With respect to the controller, although many regulators exist, like the LQR or minimum variance control, the PID controller is, by far, the most used algorithm for this application. It behaves reasonably well as long as the requirements on performance are not too high. It is also easy to tune if the parameters of the plant vary over time and in addition, several standard procedures to do so are available in the literature. In general, for systems with a first or second order dynamic, a PID controller is more than enough [1]. It also has the advantage that is easy to implement in a microprocessor as several PID control libraries are already available.

In the case of electrical drives, if we use a cascade control scheme, the dynamic seen by each loop is essentially of first order, therefore the use of a PI controller, i.e. without the derivative part, is more than enough to achieve a zero steady state error and an adequate transient response.

3.1 Current Loop

Any electrical machine is composed by an electrical part that generates the electromechanical torque required to drive the mechanical part. In a typical drive, a simplified electrical transfer function is a first order system composed by a resistance and an inductance. The form of such transfer function is:

$$P_i(s) = \frac{\frac{1}{R_s}}{\frac{L_s}{R_s}s + 1} \tag{3.1}$$

A typical PI controller has two tuneable parameters: the proportional gain K_p and the integration time T_i. The transfer function is then:

$$PI(s) = K_p(1 + \frac{1}{T_i s}) \tag{3.2}$$

Two methods will be proposed to tune this loop.

3.1.1 Pole Cancellation

By rearranging Equation (3.2) as follows:

$$PI_i(s) = K_p(\frac{T_i s + 1}{T_i s}) \tag{3.3}$$

we can see clearly that the PI controller adds a zero and a pole to the open loop transfer function. By a proper selection of those parameters we can aim to cancel the dynamics of the electrical transfer function. In fact, if we write the open loop transfer function:

$$L_i(s) = K_p(\frac{T_i s + 1}{T_i s})\frac{\frac{1}{R_s}}{\frac{L_s}{R_s}s + 1} \tag{3.4}$$

choosing $T_i = \frac{L_s}{R_s}$ eliminates the effect of the electrical pole and we end with a pure integrator open loop transfer function as:

$$L_i(s) = \frac{K_p}{L_s s} \tag{3.5}$$

Selecting the desired crossover frequency ω_c, the proportional gain is set by:

$$K_p = L_s \omega_c \tag{3.6}$$

3.1.2 Pole Placement

Another approach instead can be to use the PI controller to move the poles of the plant to specific places. The current close loop transfer function is:

$$CL_i(s) = \frac{PI_i(s)P_i(s)}{1 + PI_i(s)P_i(s)} = \frac{\frac{K_p}{L_s}s + \frac{K_p}{T_i L_s}}{s^2 + s(\frac{R_s}{L_s} + \frac{K_p}{L_s}) + \frac{K_p}{T_i L_s}} \tag{3.7}$$

where the denominator is called the characteristic polynomial and it has the typical form of a second order system:

$$s^2 + s(\frac{R_s}{L_s} + \frac{K_p}{L_s}) + \frac{K_p}{T_i L_s} = s^2 + 2\xi\omega_n + \omega_n^2 \tag{3.8}$$

Therefore, by comparison: $\omega_n^2 = \frac{K_p}{T_i L_s}$, $2\xi\omega_n = \frac{R_s}{L_s} + \frac{K_p}{L_s}$. We want a system with fast response without oscillations. Considering a critically damped behavior, $\xi = 1$ and the values of the PI controller are given by:

$$T_i = \frac{2}{\omega_n} - \frac{R_s}{L_s \omega_n^2} \quad ; K_p = 2L_s\omega_n - R_s \tag{3.9}$$

and the desired crossover frequency is simply set by saying $\omega_n = \omega_c$.

3.1.3 Comparison Between Controllers

To test the behavior of the tuning methods, we will run a series of simulations to observe the behavior of the current loop and conclude about the best method. The motor to control is the one described on Table 4.4. The desired bandwidths are $\omega_c = 1256 rad/sec$ and $\omega_{cs} = 251 rad/sec$.

Figure 3.1 and 3.2 shows a comparison betwen the Bode diagramas and step responses respectively. It can be seen that the pole cancellation has a lower overall bandwidth with respect to pole placement, however, pole placement has an overshoot in the response typical because it was being designed as a critically damped system.

In general, it looks like the pole cancellation method behaves better than pole placement. The problem is that, in reality, the cancellation method is not feasible because it needs an a priory knowledge of the parameters of the plant without any variations. So, if for example, the resistance in the motor changes, and it does it over time, then the method does not cancel the dynamics of the system but instead adds a zero very close to a pole, generating a very fast rising time but a longer stabilization time, that depends on the difference between the pole and the zero frequencies close to each other. It is due to this problem that we have decided to use the pole placement method for our current loop design.

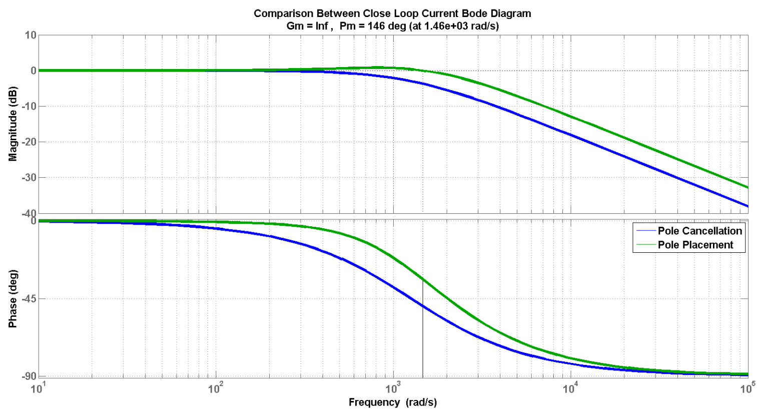


Figure 3.1: Comparison between current loop tuning methods

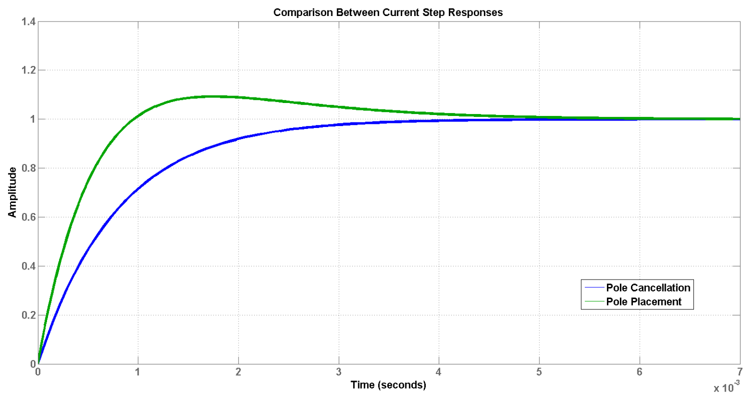


Figure 3.2: Comparison between step responses for two different tunings

3.2 Speed Loop

Once that the current loop is designed, we can substitute it for a block called $LI(s)$ and then, we can consider the speed loop diagram in Figure 3.3, where the mechanical transfer function is composed by the rotor's inertia J and the friction of the axle B. Considering that the friction is negligible, the mechanical transfer function becomes a pure integrator system.

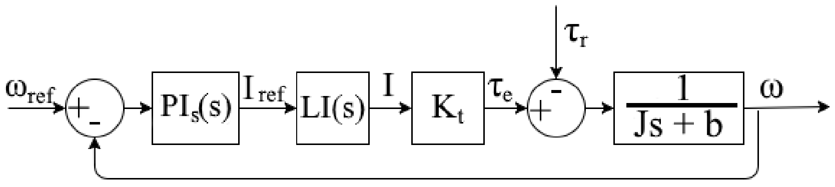


Figure 3.3: Speed control loop.

The transfer function considered is then:

$$P_s(s) = \frac{1}{J_m s} \tag{3.10}$$

Considering another PI controller like:

$$PI_s(s) = K_{ps}(\frac{T_{is}s+1}{T_{is}s}) \tag{3.11}$$

we can consider two different design methods: Low Frequency Zero and Symmetrical Optimum.

3.2.1 Low Frequency Zero

Assuming that $LI(s)$ is faster than the speed loop and damped, we can write the open loop transfer function for the speed as:

$$L_s(s) = \frac{K_{ps}K_t}{T_{is}s}\frac{T_{is}+1}{Js} \tag{3.12}$$

We can see that the system contains a double integrator, therefore pole cancellation is not feasible because a zero in the origin would create an unstable system.

Considering a desired crossover frequency ω_{cs}, we can select $T_i = \frac{1}{0.1\omega_{cs}}$, so as to generate a low frequency zero. We can approximate $L_s(s)$ around the crossover frequency as:

$$L_s(s) \approx \frac{K_t K_{ps}}{J_m s} = \frac{\omega_{cs}}{s} \tag{3.13}$$

The proportional gain will give us the desired bandwidth:

$$K_{ps} = \frac{\omega_{cs} J}{K_t} \tag{3.14}$$

3.2.2 Symmetrical Optimum

A more complex method is taking into account the effect of $LI(s)$ but instead to approximate it as a first order system. Therefore the open loop transfer function to consider is:

$$L_{so}(s) = \frac{K_{ps} K_t}{T_{is} s} \frac{T_{is} + 1}{Js} \frac{K_i}{T_{ai} s + 1} \tag{3.15}$$

where K_i and $T_{ai} = 1/\omega_c$ are the gain and time constant of the current loop respectively. The idea is then to select the crossover frequency ω_{cs} so as to be in the geommetrical mean of the two corner frequencies $w_c = 1/T_{ai}$ and $w_{zs} = 1/T_{is}$ in order to obtain the maximum phase margin possible. Graphically, the idea is shown in Figure 3.4 and hence the name of Symmetrical Optimum method [10] comes from the fact that the bode is symmetrical with respect to the crossover frequency.

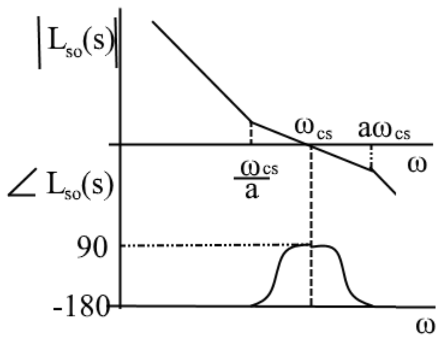


Figure 3.4: Symmetrical Optimum Method graphically explained

We can select $T_{is} = a^2 T_{ai}$ with $a > 1$. Then, the crossover frequency of the speed loop is:

$$\omega_{cs} = \frac{1}{\sqrt{T_{is} T_{ai}}} = \frac{1}{a T_{ai}} \tag{3.16}$$

and the proportional gain of the controller is:

$$K_{ps} = \frac{1}{a K_t K_i} \frac{J}{T_{ai}} \tag{3.17}$$

The poles of the close loop transfer function are a complex pole pair of the form:

$$s = w_{cs}[-D \pm j\sqrt{1-D^2}] \tag{3.18}$$

being $D = \frac{a-1}{2}$ the damping factor of the system.

3.2.3 Comparison Between Controllers

Considering again the motor described in Table 4.4 an the desired bandwidths $\omega_c = 1256 rad/sec$ and $\omega_{cs} = 251 rad/sec$. We will compare the tuning methods described above. The results can be seen on Figures 3.6, 3.5 and 3.7.

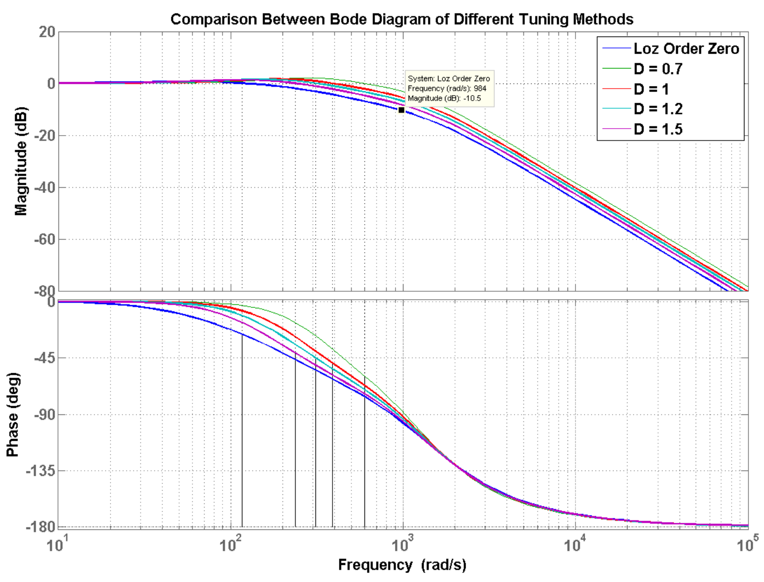


Figure 3.5: Comparison between the Bode diagram for different tuning methods and different dampings.

We can see in Figure 3.6 that the *Low Frequency Zero* has the lowest settling time of all the methods, but it also has the smallest overshoot. For the *Symmetrical Optimum approach*, we can see that is depends highly on the damping factor D.

Figure 3.5 shows the Bode diagram of the simulations in which it can be seen that the smallest bandwidth comes from the *Low Frequency Zer* , but in the others, the bandwidth changes with the damping factor, which confirms the step response results.

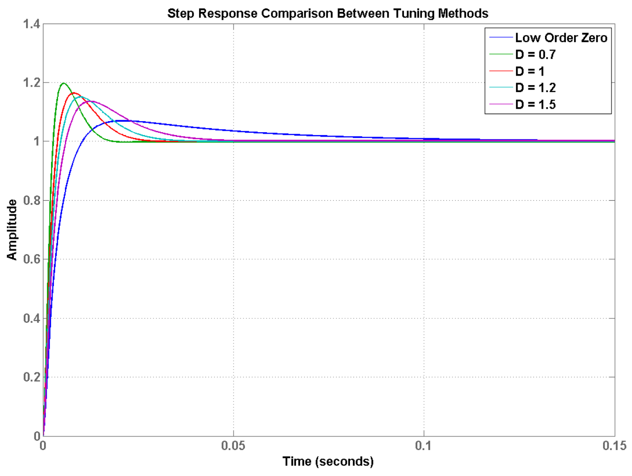


Figure 3.6: Comparison between step responses for different tuning methods

The load disturbance rejection sensitivity function in Figure 3.7 shows, that for any of the cases of the *Symmetrical Optimum*, the system rejects the load much better than the *Low Frequency Zero*. This behavior is highly relevant because in real life, we will be dealing permanently with torque disturbances due to the dynamics of the vehicle.

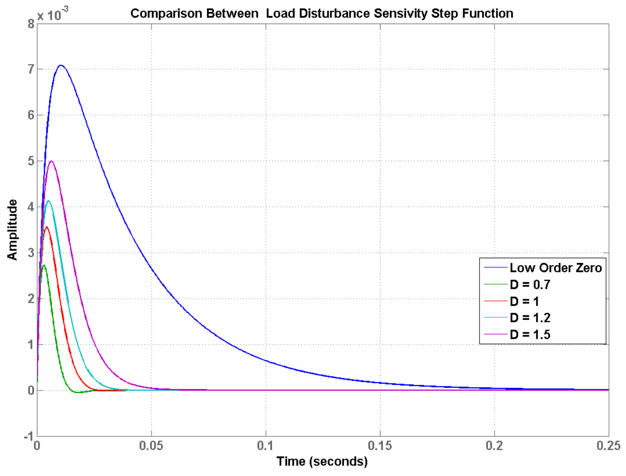


Figure 3.7: Comparison between load disturbances sensitivity function for different speed loop tuning methods

Considering all the above, we can see that the *Symmetrical Optimum* approach gives in better results and therefore is the tuning method chosen.

Chapter 4

Simulation and Comparison

4.1 Introduction

The simulation of the model depicted in Chapter 2 was performed using MATLAB & Simulink. Its main objective is the verification of parameters describing the operation of the vehicle. Simulations take into account the following variables to conclude about the accuracy of the model:

- Position in the L plane of the vehicle: seeks to verify that the vehicle follows the trajectory specified as input parameter of the model.
- Kinematic variables: X-axis velocity, X-axis acceleration, angular velocity and angular acceleration $\dot{\psi}$. These variables seek to verify the dynamic model of the vehicle, at the same time, they are necessary to obtain the vehicle position.
- Dynamic variables: all the forces applied to the vehicle. They are useful to validate the dynamic behavior of the system.
- Electrical variables: it includes the torques generated by the motors which are the inputs for the vehicle dynamics block. Current, voltage and power of each motor make also part of these variables. Finally, it includes the energy consumption for a specific simulation.
- Input parameters: in order to control that there are no errors in the inputs during the simulation, we monitor the reference speed and the torque generated by the motors.

The equivalent dynamic model of the DC brushless motor implemented in Simulink is shown in Figure 4.1. The speed constant K_T and the torque constant K_e are given in the datasheet and their values are equivalent. In the block *Wheel Inertia* is implemented the motor's mechanical transfer function as stated in Equation (2.43).

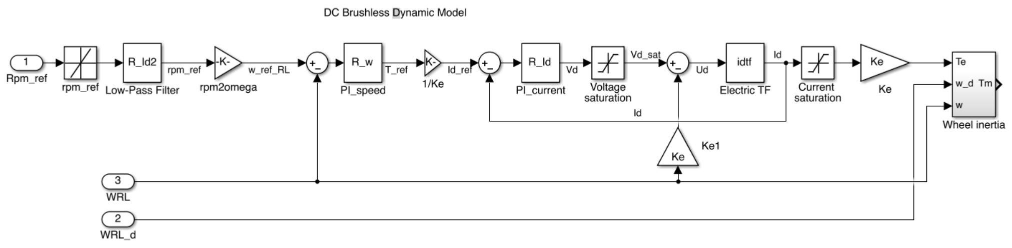


Figure 4.1: Complete Simulink model of the motors.

The saturation voltage is taken as the nominal voltage of the source. The saturation current is calculated considering the maximum current admissible when the motor is delivering the maximum torque at stall T_{stall}. Using the relation in Equation (4.1) is possible to obtain the value.

$$I_{sat} = \frac{\tau_{stall}}{K_e} \tag{4.1}$$

The model also includes blocks for the signal processing of the speed reference: it corresponds to a proper saturation and the respective conversion from revolutions per minute (rpm) to radians per second (rad/s).

In addition to the inner current loop and the mechanical loop, in the block *Low-Pass Filter* a regulator is implemented in order to adjust the bandwidth with respect to the perturbations without changing the total bandwidth. This will be explained in more detail in Section 4.2.3

4.2 Control Loops Simulation

4.2.1 Current Loop Simulation

The bandwidth in this kind of motors, is usually high (in the order of thousands of rad/s depending on the application), but in particular the model *HUB10GL* has a time constant of $4.75ms$ which is low with respect to the typical values. Designing a higher value could be problematic because the control action would not take place. The electrical transfer function is given in Equation (3.1).

Table 4.1 shows the data concerning the closed-loop step response in Figure 4.2, there the selected controller is already implemented. The overshoot is a characteristic of the chosen design method.

Parameter	Value	Units
Rise Time	0.0014	s
Settling Time	0.0066	s
SettlingMin	0.9122	-
SettlingMax	1.1246	-
Overshoot	12.4586	-
Undershoot	0	-
Peak	1.1246	-
Peak Time	0.0034	s

Table 4.1: Step response information of the closed-loop transfer function.

Figure 4.2 shows a simulation in Simulink of the loop implementing the control action. Figure 4.1 contains in the inner loop the function *electric tf*, which is the transfer function used during the design.

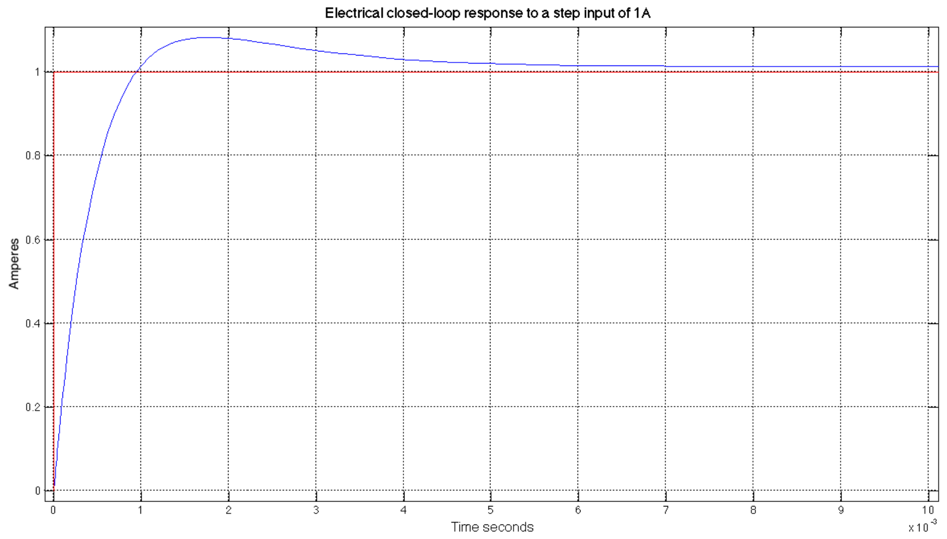


Figure 4.2: Electrical closed-loop response to a step input of 1A (Simulation).

The simulation consists in a constant step input with amplitude of 1 A. It also verifies the data given in Table 4.1, since the values in 4.2 are close to the real ones.

Parameter	Value	Units
Rise Time	0.001	s
Settling Time	0.006	s
SettlingMin	1	-
SettlingMax	1.11	-
Overshoot	11.1	-
Undershoot	0	-
Peak	1.11	-
Peak Time	0.0021	s

Table 4.2: Step response information of the closed-loop transfer function (Simulink data).

4.2.2 Speed Loop Simulation

In the design of this controller, we consider that the current loop is designed to be much faster than the mechanical dynamics. Due to the difference in velocity of the loops, we consider only the mechanical specifications. The mechanical transfer function is given in Equation (3.10).

The resultant parameters of the PI controller are:

$$K_p = J_m \times \omega_c = 3.344 \tag{4.2}$$

$$\frac{1}{T_i} = \omega_c \times 0.1 = 18.85 \tag{4.3}$$

Parameter	Value	Units
Rise Time	0.0094	s
Settling Time	0.0934	s
SettlingMin	0.9174	-
SettlingMax	1.0697	-
Overshoot	6.9673	-
Undershoot	0	-
Peak	1.0697	-
Peak Time	0.0281	s

Table 4.3: Step response information of the closed-loop transfer function.

In this way, the loop transfer function can be approximated around its crossover frequency by its high frequency approximation. Table 4.3 shows the data concerning the step response of the closed-loop transfer function with the controller implemented.

Now, we present a Simulink experiment in which the vehicle dynamics is present in the loop. The simulation in Figure 4.3 is the response of the system to a unitary rpm step.

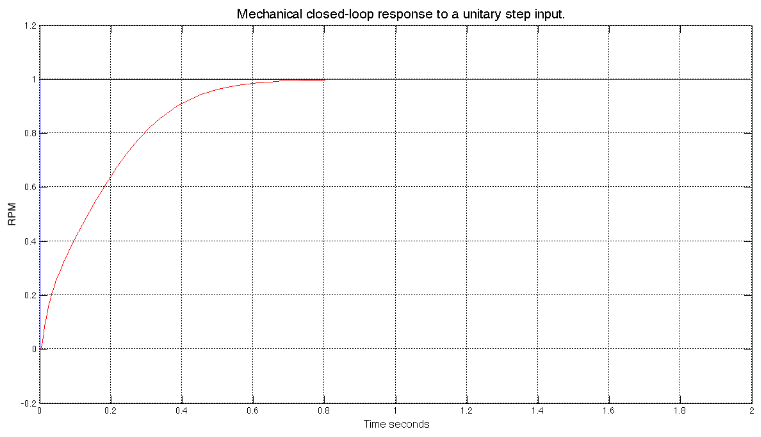


Figure 4.3: Mechanical closed-loop response to a unitary step input.

4.2.3 Low-Pass Filter Simulation

Due to the slow dynamics of the electrical system for this motor, it was necessary to implement a low-pass filter [15] before the speed loop control so as to add another degree of freedom to the system. Although the bandwidth of the current loop was above the recommended limit, it is advisable to implement this filter to increase the perturbations rejection in the inner loop. Additionally, this filter is useful to reduce system noise, which makes it a convenient implementation.

The new bandwidth value we chose is $251.3 rad/s$ and for this purpose a filter was used as follows:

$$H(s) = \frac{1}{\frac{s}{\omega_c} + 1} \tag{4.4}$$

where ω_c is the value that we need to design.

Figure 4.4 shows the Bode diagram of the complete system with the mentioned implementation. The resulting bandwidth is $275.83 rad/s$, which compared to the initial value meets better the requirements.

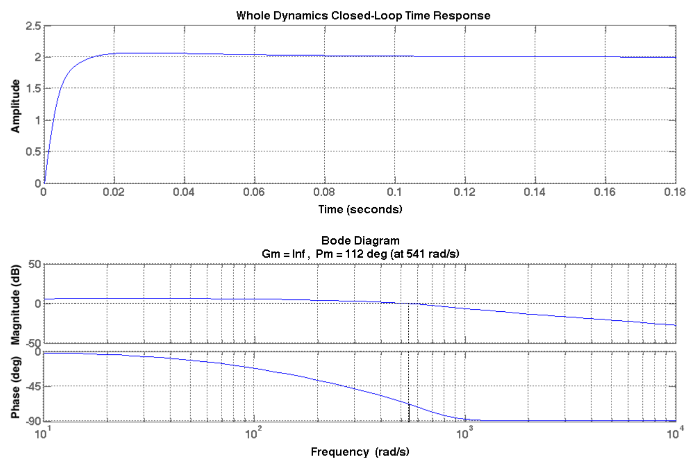


Figure 4.4: Total system response after low-pass filter implementation.

4.3 Model Simulation in Simulink

The DC brushless motor used for the model is the HUB10GL from the brand *Uumotor* and it includes the wheel dynamics, therefore the moment of inertia is the one of the rotor added to the one of the wheel. Specifications appears in the Table 4.4.

Some parameters such as the no-load speed, the no-load current and the speed-torque gradient are not specified in the datasheet, these data must be obtained in the laboratory.

The main goal of this section is the assessment of the mechanical torque produced by the motors, which will feed the *Vehicle dynamics* block. In order to validate this parameter, the vehicle follows two trajectories in which the conditions of the road are set according to the permitted limits. The two input paths have the following features:

1. Straight line with zero degrees slope: the vehicle must reach the maximum speed with a constant acceleration, which is also maximum. The nominal power and nominal torque must not be exceeded. Similarly, the stall current and stall torque will not be overcome.

2. A couple of curves with constant amplitude. In addition, the trajectory has a constant slope of 45 degrees: under these conditions the vehicle will face conditions in which appears the stall current and stall torque due to the duration of the climb.

Table 4.4 shows the data specification of the HUB10GL motor. By using these data it is possible to compare the information obtained from

Parameter	Value	Units
Nominal Voltage	36	V
Nominal Power	500	W
No Load Speed	-	rpm
No Load Current	-	mA
Nominal Speed	750	rpm
Nominal Torque	6.5	Nm
Nominal Current	13.89	A
Stall Torque	20.02	Nm
Starting Current	55.56	A
Terminal Resistance phase to phase	0.160	ohms
Terminal Inductance phase to phase	0.76	mH
Torque Constant	0.03775	Nm/A
Speed Constant	0.03775	V/rpm
Speed-Torque gradient	-	rpm/Nm
Mechanical Time Constant	77	ms
Rotor Inertia	0.0177	Kgm^2
Number of pole pairs	10	-
Number of phases	3	-
Weight of the Motor	3.5	Kg

Table 4.4: HUB10GL motor data specifications.

the simulations. Similarly, the assessment of parameters such as nominal power, nominal torque, stall torque and starting current is included in this analysis. On the other hand, the evaluation of energy consumption depends on the trajectory: an estimate of consumption is done for the conditions specified in every simulation.

4.3.1 Trajectory 1

Figure 4.5 shows that the simulation takes 20 seconds. In addition, the trajectory has a length of 175 meters in the x-axis and zero length in the y-axis. The modeled trajectory generator is in charge of the construction of the desired path.

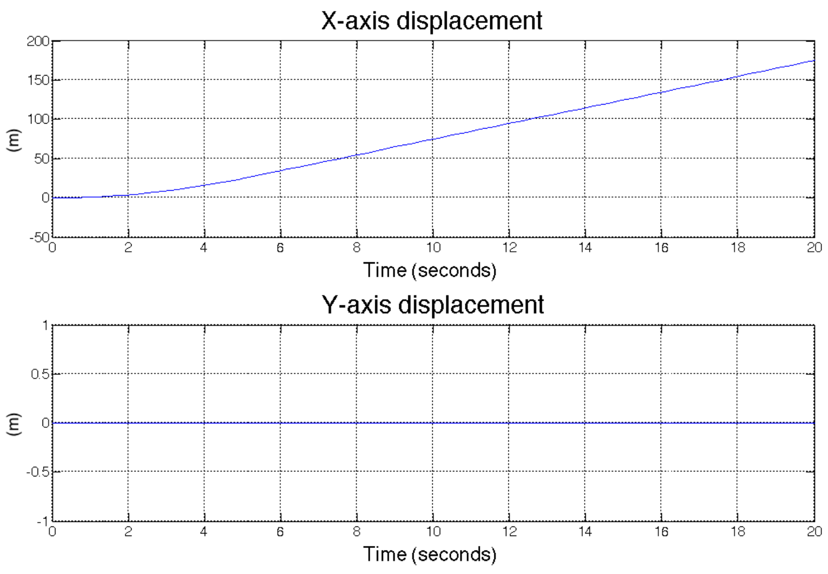


Figure 4.5: Experimental trajectory (1).

Motor Measurements

First, Table 4.4 shows that the stall torque supplied by the motors is $20.02Nm$, In Figure 4.6 the maximum torque is $7.42Nm$ for two wheels ($3.71Nm$ per wheel), then the simulation meets the specifications for this parameter.

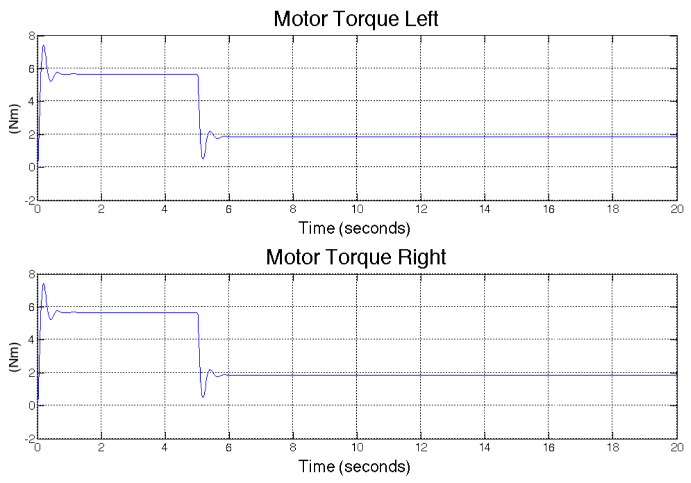


Figure 4.6: Output torque of the motors (Trajectory 1).

Secondly, the nominal torque is $6.5Nm$ according to the datasheet. Figure 4.6 shows that the torque does not exceed this value in the simulation.

In fact, the nominal torque for this simulation was $0.94Nm$ per wheel. On the other hand, Figure 4.7 shows the time history of the speed reference.

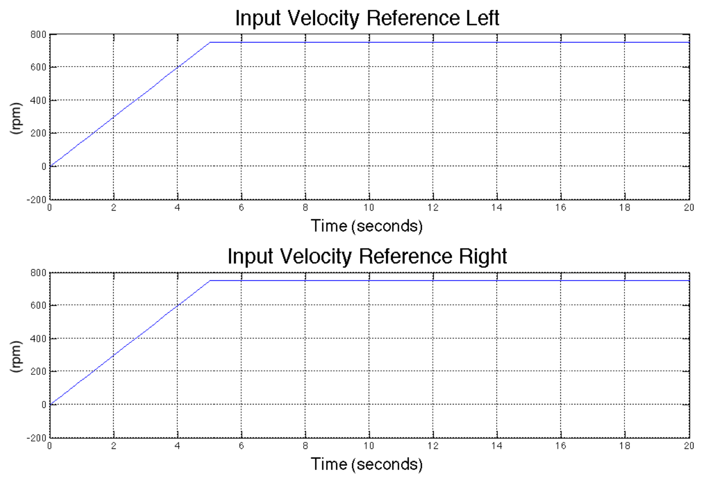


Figure 4.7: Input velocity reference (Trajectory 1).

Now, as the torque value is inside the desired range, we expect that the current, voltage and power parameters have correlation to these results. Figure 4.8 shows that the maximum current is $43.5A$ for two motors ($21.75A$ per motor). As in the datasheet is reported $55.56A$, the parameter is well working.

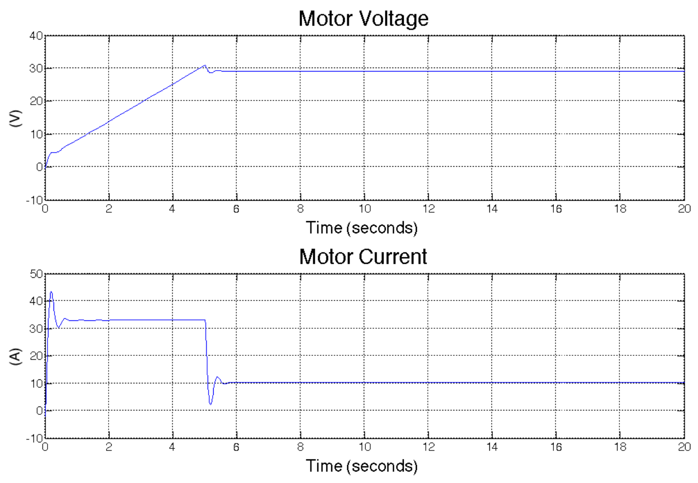


Figure 4.8: Motor measurements: current and voltage (Trajectory 1).

Finally, the electric power is calculated by using the nominal voltage and the required current at each time instant. Figure 4.9 shows the time history of the power, it can be verified that the motors are working in normal conditions because they do not overcome the nominal power of $500W$ in steady state condition.

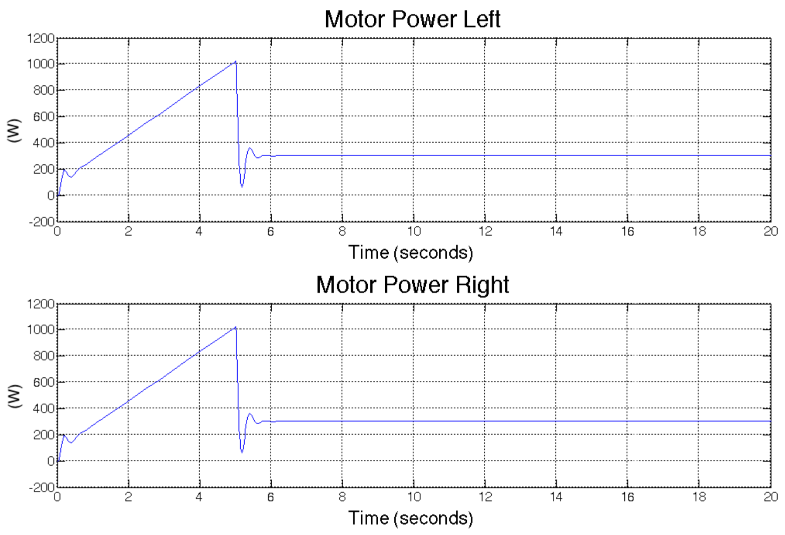


Figure 4.9: Motor measurements: power (Trajectory 1).

In order to calculate the power consumption, it is enough with the integral of the time history of the electric power consumed by each motor giving us the energy consumption shown in Figure 4.10. After that, the sum of these values will result in the total energy consumption for this trajectory. After a proper conversion from watts per second to watts per hour, the energy for the simulation under these conditions is $733Wh$ per one our running.

The capacity of the batteries implemented on the vehicle is $972Wh$, which permits to conclude that the vehicle will have an autonomy in this situation of 79 minutes.

Vehicle Dynamics Simulation

From the kinematic point of view, the verification of the model of the skid-steering robot is important not only to guarantee that it is working well, but also to make available the variables required to generate the local variables and the resulting trajectory. It is necessary to assess variables such as: X-axis velocity, X-axis acceleration, angular velocity and angular acceleration. Table 4.4 reports the nominal speed and the maximum acceleration is $2m/s^2$. The results for the trajectories that were stated are listed below.

Figure 4.11 shows the variables time history. The maximum X-axis ve-

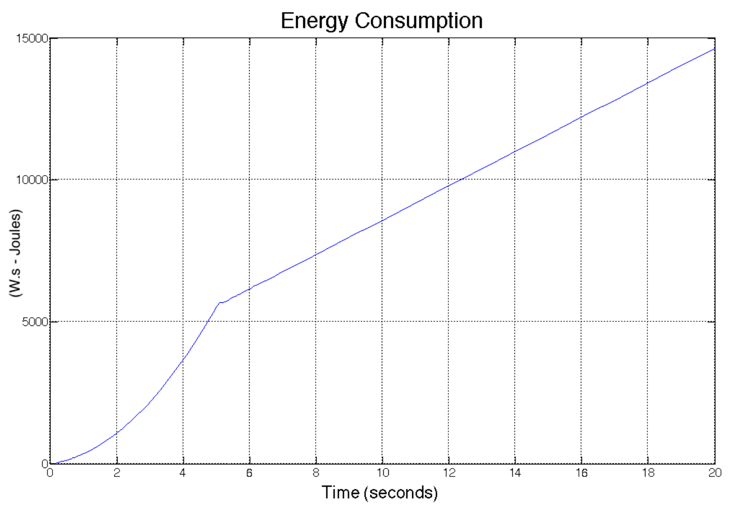


Figure 4.10: Power consumption in watts per second (Joules) (Trajectory 1).

locity that the vehicle develops in this case is $10m/s$ and the maximum X-axis acceleration is $2m/s^2$. These data coincide with the specifications of the vehicle and as expected, the values are inside the desired range for a simulation in which we predicted that the maximum values would appear, as there are no constraints on the vehicle.

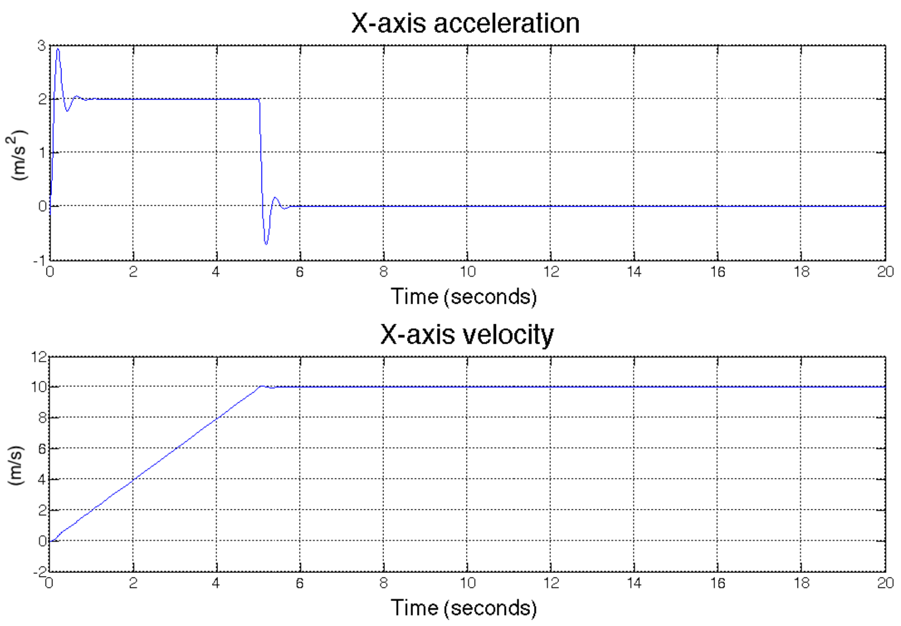


Figure 4.11: Kinematic variables (1).

In Figure 4.11, the acceleration increases until its maximum value, when this occurs it is constant until the maximum velocity is reached. With maximum velocity the acceleration becomes zero, from that instant, the only variation is to provide a negative acceleration until the vehicle stops and finally, both velocity and acceleration are zero.

Figure 4.12 shows that the angular velocity and angular acceleration are zero. In this case, the vehicle follows a straight path in the x-axis, so any variation of these variables was expected.

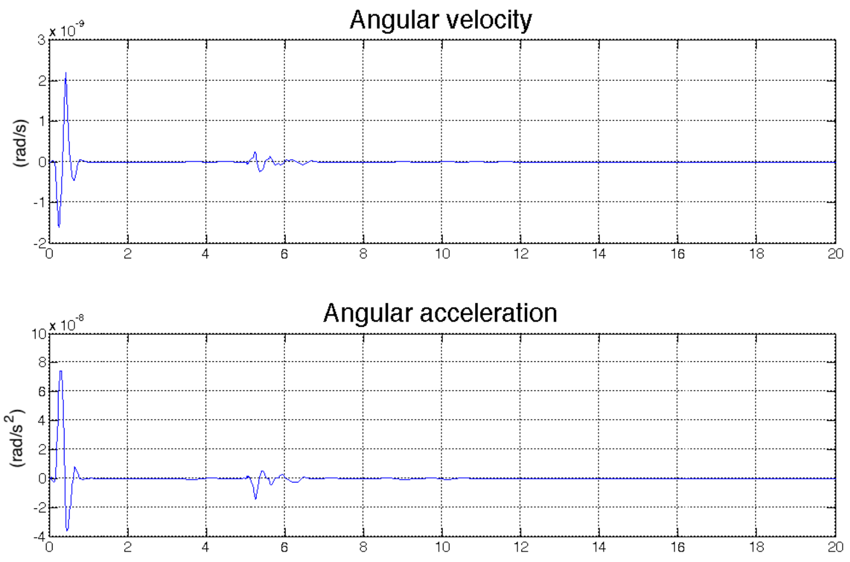


Figure 4.12: Kinematic variables (2).

From the dynamic point of view, the simulation was focused more towards a verification of the equations that were stated for the system dynamics. The variables that we are testing are: longitudinal resistive force, normal force, transversal resistive force and resistive moment of inertia.

Figure 4.13 shows the resultant normal forces in the model. As this is a straight path without slope, the expected result is that these values are equivalent for the four wheels. Indeed, the value of the normal forces is constant for the whole simulation and equal on each wheel.

Regarding the transversal resistive force, the longitudinal resistive force and the resistive moment of inertia of the vehicle, these are described in Figure 4.14. First, for this trajectory we do not expect variations neither in the trasnversal force nor in the moment of inertia, as the vehicle is following a straight path in x-axis. Figure 4.14 permits to easily check this information. On the other hand, the longitudinal force is a constant line as there are no variations in the characteristics of the ground.

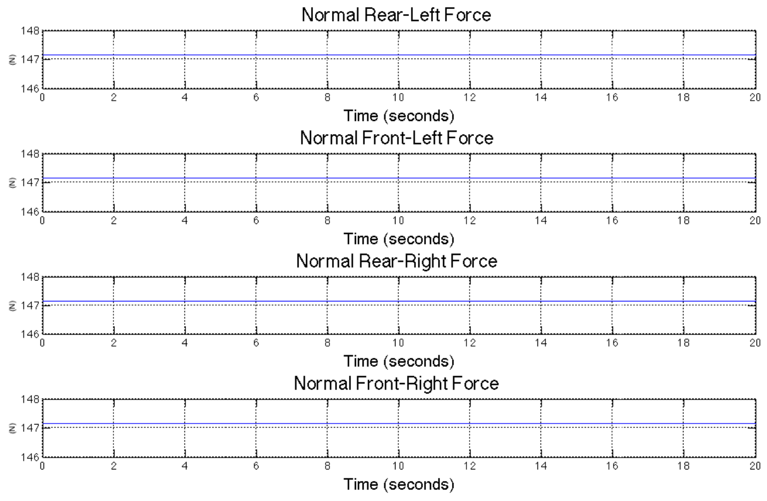


Figure 4.13: Resultant normal forces of the vehicle (Simulation 1).

XY Plane Results - Trajectory Tracking

After the implementation of the transformation block and using as inputs the X-axis velocity and angular position, it is possible to obtain the variables in the local reference frame. Obtaining these variables is a essential part in the simulation process because that would allow us to know the response of the system fed directly with a given trajectory. At this point, it would be possible to compare the desired trajectory against the generated one by only using the output variables of the dynamic model.

For this simulation was used as input the path shown in Figure 4.5, the output of the block implementing the global to local transformation traces the results generated by the model of the skid-steering robot. Figure 4.15 displays the result of the simulation for these conditions, one can observe in the input vs output comparison that the result is very accurate and the error is almost negligible. This leads to the conclusion that the reconstruction was successfully implemented. Note that in Figure 4.15, the red graph is hiding the blue one, so no differences between input and output are distinguished.

4.3.2 Trajectory 2

The idea was to recreate the behavior of the vehicle as it was driving through a vineyard. It can be seen in Figure 4.16 that the simulation takes 100 seconds. The vehicle performs a curve with an amplitude of 100 meters in the y-axis, while in the x-axis it advances 10 meters per each complete curve.

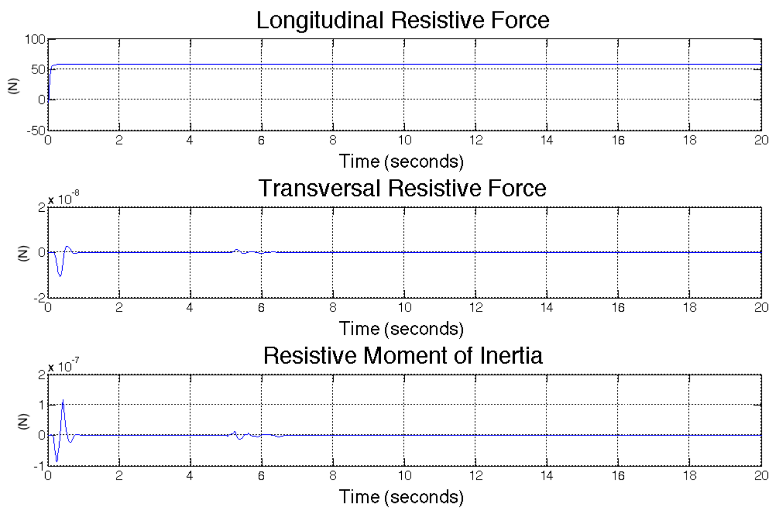


Figure 4.14: Resistive forces (Simulation 1).

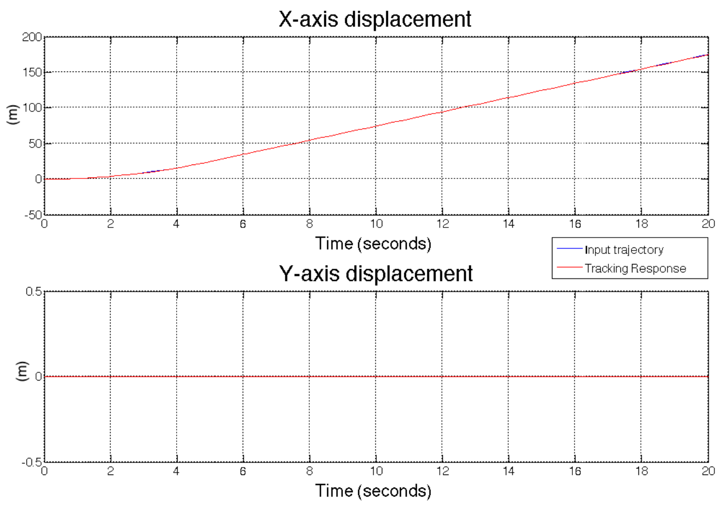


Figure 4.15: Result of the tracking of the parameters (Simulation 1).

Motor Measurements

Figure 4.17 shows the mechanical torque time history. In this case, the maximum value is $16Nm$ ($8Nm$ per wheel) and it appears when the vehicle reach the maximum speed. When it happens, the vehicle starts to decelerate until its speed is near zero at the end of the curve, making the turn easier and preventing the motors to feed torques above the nominal value for long time periods. As the vehicle follows this behavior in each curve, the torque has an alternating-wave behavior around a positive constant value.

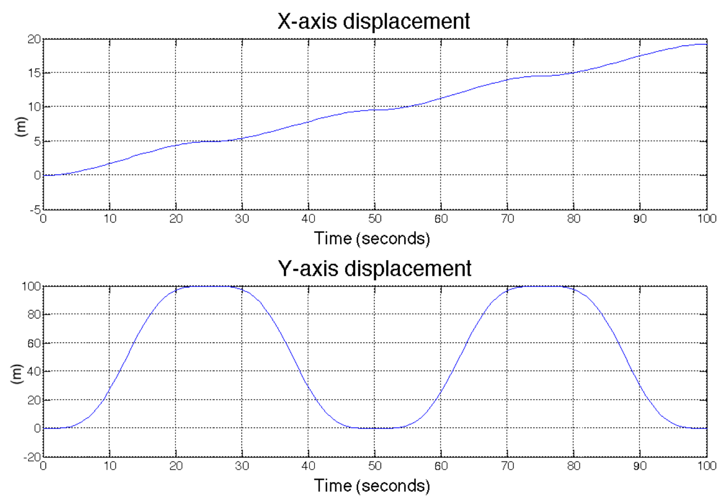


Figure 4.16: Experimental trajectory (2).

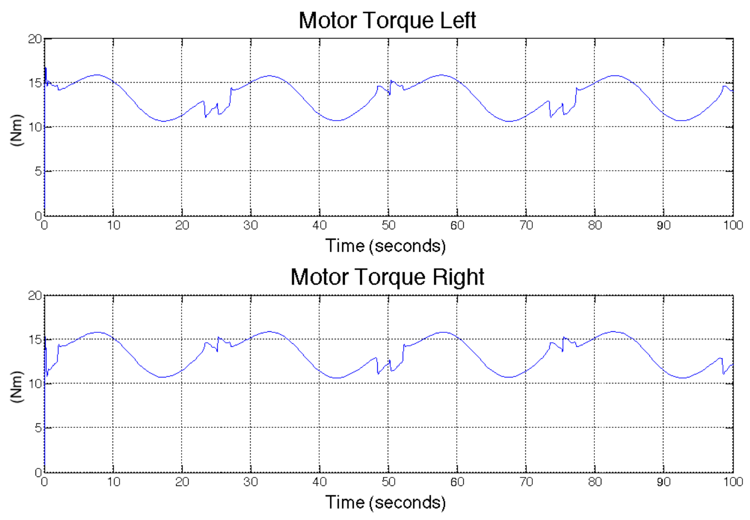


Figure 4.17: Output torque of the motors (Trajectory 2).

The maximum time in which the motors provide torque above the nominal value is 10 seconds and the stall torque is not exceeded during the simulation. Thus, we can confirm that the design works well with a 45 degrees slope. Figure 4.18 shows the input velocity reference.

Figure 4.19 shows the results for current and voltage. Here we see that the stall current is not exceeded, as the maximum value is $89A$ for one side of the vehicle, that means $44.5A$ per wheel. The experimental nominal current in which the motors work is high but normal for this simulation. Therefore, it is advisable to take trajectories with these characteristics at low speed or only for short periods of time. In addition, it is necessary to know how long the motors are able to supply currents above the nominal value to avoid permanent damage to the structure.

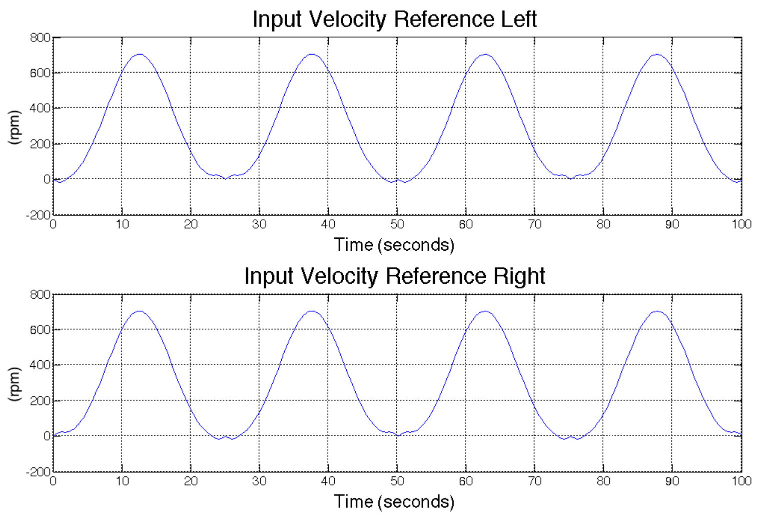


Figure 4.18: Input velocity reference (Trajectory 2).

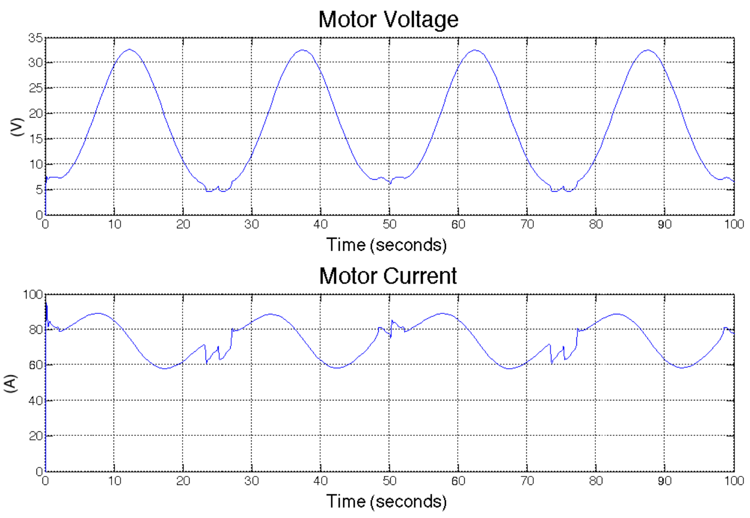


Figure 4.19: Motor measurements: current and voltage (Trajectory 2).

On the other hand, Figure 4.20 illustrates the power. It faces the same conditions described for the current, then similar conclusions can be stated. The average power is high and although the trajectory is not performed at maximum speed, this value requires verification of the time during which the motors can supply values above the nominal.

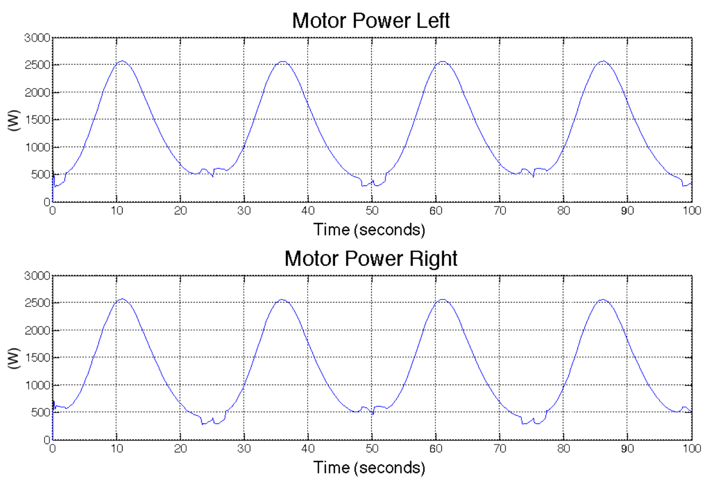


Figure 4.20: Motor measurements: power (Trajectory 2).

Finally, in Figure 4.21 the time history of the energy consumption is shown. which by calculations allows us to conclude that the energy consumption here in type of paths per hour is $2570Wh$. Recalling that the battery capacity is $972Wh$ we can conclude that the vehicle will have an autonomy of 23 minutes approximately. It is important to recall that these are extreme conditions with a slope of $45\ \angle$, which in reality it can not sustain this situation for that amount of time due to overheating of the motors.

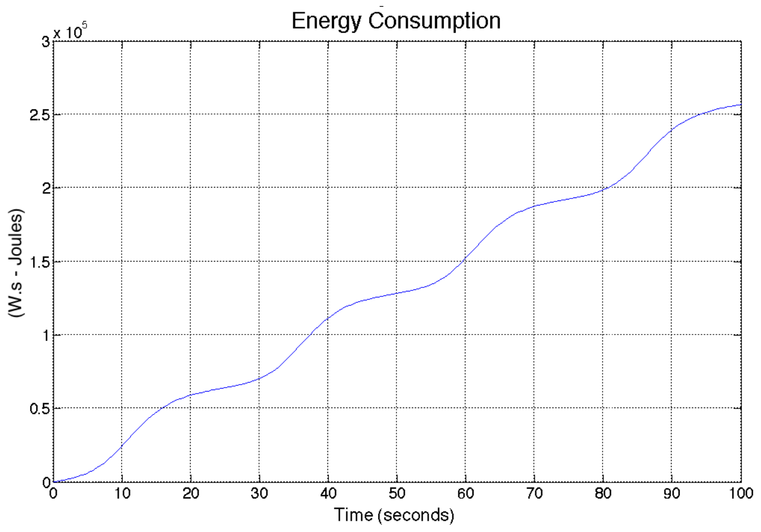


Figure 4.21: Power consumption in watts per second (Joules)(Trajectory 2).

Note: the current and torque in the vehicle will exceed nominal values when the slope increases, but only for short time intervals. We do not perform other simulations in the subject because in experiments one could see the motor driving these values for an unlimited time, but in reality this condition cannot endure for long periods of time due to motor heating.

Vehicle Dynamics Simulation

Figure 4.22 depicts the resulting kinematic variables for this trajectory. For the first part of the simulation, the behavior is similar to the first experiment, with the difference that the slope causes the vehicle to take longer to reach the reference speed. Similarly, the X-axis acceleration keeps the behavior registered for the first experiment, but now, its value switches to negative acceleration when maximum speed is reached. After that, the vehicle decelerates to almost zero speed before facing the next curve. It was decided that the velocity profile will be held in this way to prevent maximum operating conditions for long periods of time.

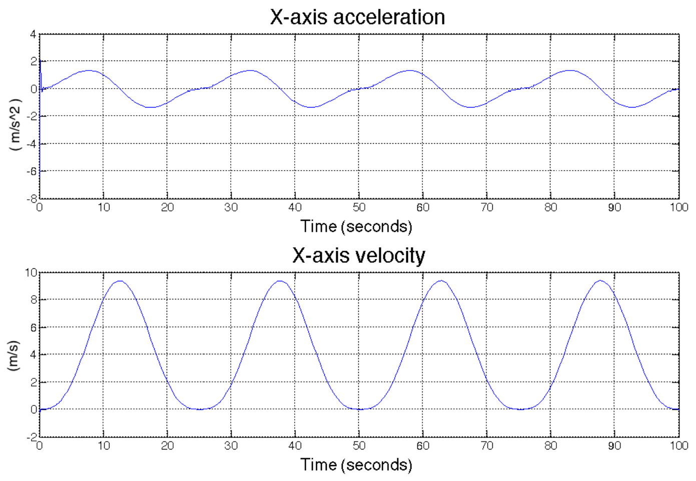


Figure 4.22: Kinematic variables (Simulation 2).

In the curves, the velocity behaves as an oscillatory wave varying from zero to 10 m/s, depending on the point on the curve where the vehicle is. The system keeps this trend until the end of the experiment.

Finally, in Figure 4.23 we can see that angular velocity and angular acceleration react as expected. There is a peak in their magnitudes at the time when the vehicle is taking the curve, after this moment, they decrease slowly to zero. At the beginning of the next curve, the values are maximum again and the behavior is repeated.

The vehicle inverts its angular velocity in each curve and as consequence the peaks of the angular acceleration are reversed to the opposite sense.

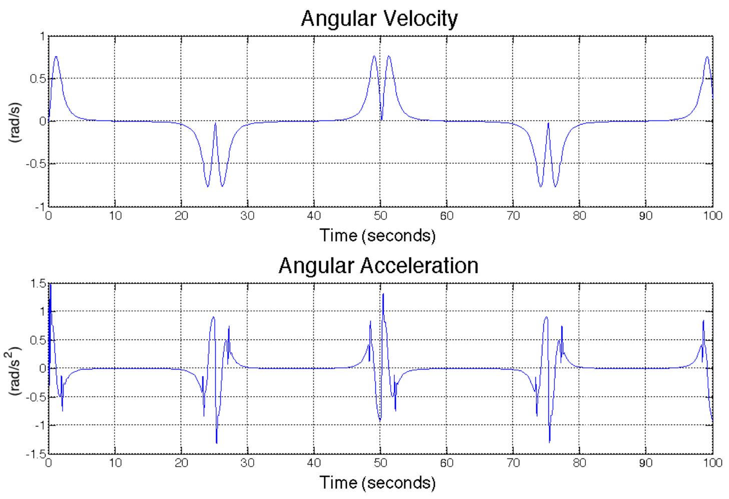


Figure 4.23: Kinematic variables (Simulation 2).

The angular velocity increases linearly until the maximum value is reached or until the vehicle changes the direction of its trajectory.

In Figure 4.24 we can observe the results from the dynamic point of view. The normal force remains constant as the orientation vector has no variations during the simulation, this was the expected result.

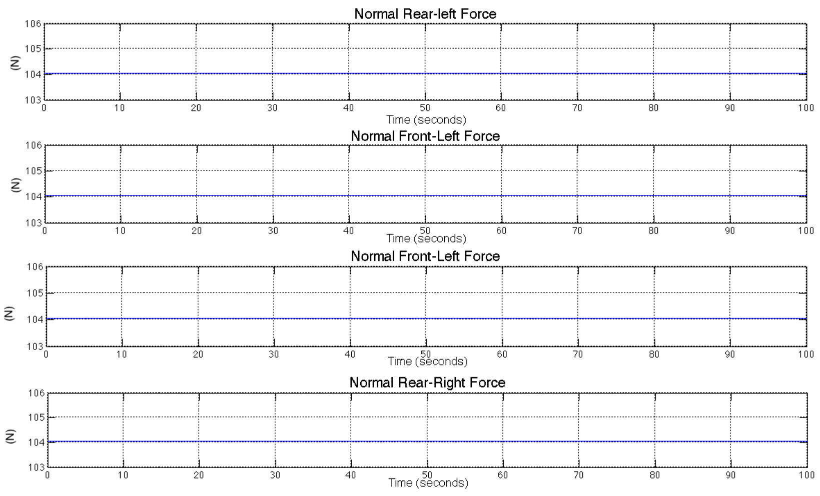


Figure 4.24: Resultant normal forces of the vehicle (Simulation 2).

On the other hand, when the vehicle faces the curve with positive slope, the longitudinal resistive force increases significantly due to the increase in the magnitude of the vector in the X-axis. The force varies during the curve, as the vehicle is constantly changing its alignment, but this oscillation is smaller because the displacement in the principal axis during the curve is not comparable with respect to the absolute displacement in its main axis. The higher the slope value, the greater the variation of this force. The trend is shown in Figure 4.25.

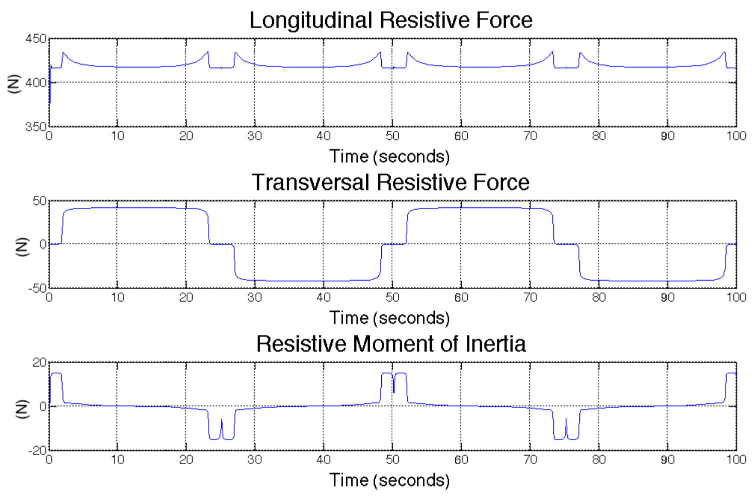


Figure 4.25: Resistive forces (Simulation 2).

For the transversal resistive force, during the curve its value is no longer zero as the component in the Y-axis is not null anymore. It begins to oscillate because the vehicle is turning around the principal axis of displacement (in the X-axis). If the amplitude of the curve increases, this vector will be increased as well.

Finally, the resistive moment of inertia is nonzero during the curve. During the turning, the resistive moment is maximum, and after that, its value decreases slowly to zero until the lateral displacement has finished. Once a new curve begins, the resistive moment of inertia is maximum again.

With these tests it is enough to confirm that the simulation of the dynamic model of the vehicle is running properly. The results are consistent with the expected ones and this fact allows us to proceed to a comparison process with a nonlinear model with a higher degree of complexity.

XY Plane Results - Trajectory Tracking

The second simulation was performed with the trajectory specified in Figure 4.16. As in the first case, the result is quite accurate and it verifies not only the operation of the transform block to local coordinates, but also the response of the complete system. Thus, one can confirm the design for each block described. Figure 4.26 shows the input vs output comparison. Apart from some differences in the transients, no notable differences between the two signals are noticed.

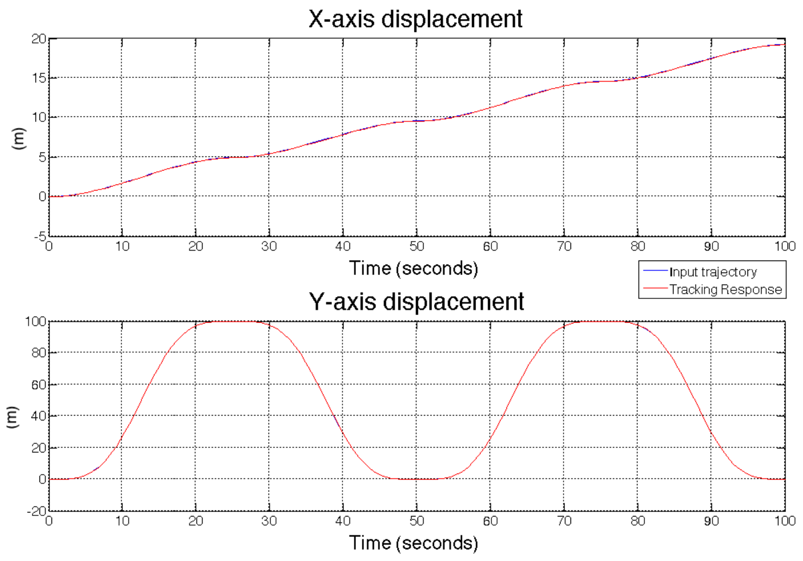


Figure 4.26: Result of the tracking of the parameters (Simulation 2).

4.4 Non-Linear Model Simulation

In addition to the evaluation that we have made in Simulink, we consider important to perform a model comparison using a model like ours, but in this case considering all non-linearities in the wheels. This model has some similarities in the vehicle dynamics analysis, but is different in the modeling of the wheels, since this implements the Pacejka magic formula which will be introduced in this section.

Model comparison will be held following these steps:

1. Adaptation of models: the first step is to insert all the parameters of our vehicle in the model, likewise, identify that the relevant variables are available in the Simulink model.

2. Simulink modeling: as in the model of a skid-steering robot, the vehicle dynamics is only a block that is part of the complete system. Therefore, the next step was to synchronize the nonlinear model with our implementations. The input is the torque generated by the motors, and the output are kinematic variables that must be transformed to the local reference frame.

3. Simultaneous simulation of models: at this point, we will simulate both models using as input the same trajectory. To perform the comparison,

we decided to use velocity and acceleration in the X-axis, angular velocity and angular acceleration. With these variables we can obtain the resulting trajectory generated by the transformation block in both models, which is our final goal.

4. Analysis and comparison: once we obtain the mentioned graphs, we will proceed to perform the analysis of results. Using the obtained data, it will be possible to conclude on the similarity of the models.

4.4.1 Pacejka 'Magic Formula' Tire Models

One of the most used non-linear approaches to tire model is the so called **Pacejka Magic Formula**, which is an equation that allows us to approximate the forces in the wheel with a reasonable accurate. However, its parameters does not have any particular physical meaning, instead, it is obtained by performing test in the wheel to obtain the curve.

The general form of the Magic Pacejka Formula is given by [13]:

$$y(x) = Dsin[CarctanBx - E(Bx - arctanBx)] \tag{4.5}$$

with $Y(X) = y(x) + S_v$ and $x = X + S_H$, where

Y : output variable F_x, F_y, or possibly M_X
X : input variable $tan(\alpha)$ or κ
B : stiffness factor
C : shape factor
D : peak value
E : curvature factor
S_H : horizontal shift
S_V : vertical shift

The formula creates a curve that pass through the origin, it grows until it reaches its maximum and then, tends to a horizontal asymptotic. The curve shows an anti-symmetric shape with respect to the origin for given coefficients B, C, D y E. Two shifts S_H and S_V are introduced in order to have an offset with respect to the origin.

Pacejka Magic Formula describes characteristics that match pretty well curves for the transversal force F_y, the longitudinal force F_x and if desired the aligning torque M_z. These are described as function of the slip angle α, the longitudinal slip κ and the camber angle γ.

Figure 4.27 introduces the typical shape of the curve, it shows also the

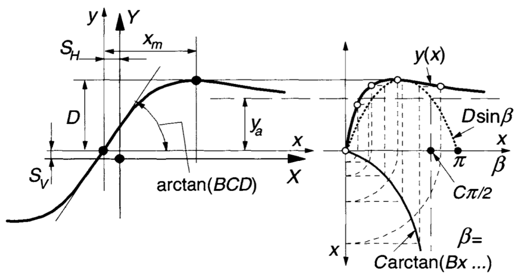


Figure 4.27: Original Pacejka magic formula (sine version).

meaning of some factors included in the formula. The description of parameters is introduced as:

- Coefficient D: it represents the peak value with respect to the x-axis. It fits for $C \geq 1$.
- Coefficient C: it regulates the limits of the Function 4.5, it is known as shape factor because defines the shape of the resulting curve. The equation describing C is:

$$C = 1 \pm (1 - \frac{2}{\pi} arcsin \frac{y_a}{D}) \tag{4.6}$$

  where y_a is illustrated in Figure 4.27.
- Coefficient B: it is the stiffness factor and is used to define the slope at the origin.
- Coefficient E: It is the parameter that controls the horizontal position of the peak and its curvature. The equation describing the value of E:

$$E = \frac{Bx_m - tan\{\pi/(2C)\}}{Bx_m - arctan(Bx_m)} \tag{4.7}$$

  where x_m is illustrated in Figure 4.27 and $C > 1$.
- Coefficients S_H and S_V: These offsets appear when conicity effects arise. For that, the rolling resistance cause Fy and Fx curves to avoid passing through the origin.

The expressions of the Pacejka coefficients to calculate F_x and F_y are:

$$\begin{cases} C_y = a_0 \\ D_y = (a_1 F_z + a_2) F_z \\ E_y = a_6 F_z + a_7 \end{cases} \tag{4.8}$$

$$\begin{cases} BCD_y = a_3 sin(2arctan(F_z/a_4))(1 - a_5\,|\gamma|) \\ B_y = BCD_y/(C_yD_y) \\ S_{h,y} = a_8\gamma + a_9F_z + a_{10} \end{cases} \tag{4.9}$$

$$\begin{cases} C_x = b_0 \\ D_x = (b_1F_z + b_2)F_z \\ E_x = b_6F_z + b_7 \end{cases} \tag{4.10}$$

$$\begin{cases} BCD_x = b_3 sin(2arctan(F_z/b_4))(1 - b_5\,|\gamma|) \\ B_x = BCD_x/(C_xD_x) \\ S_{h,x} = b_8\gamma + b_9F_z + b_10 \end{cases} \tag{4.11}$$

Each parameter of the *Pacejka '89 specification* [14] affects the resulting curve, in order to properly estimate these parameters, it must be performed a set of experiments in the tire we are interested in.

Pacejka 'Magic Formula' Implementation

The experimental parameters to be implemented in the Pacejka magic formula are given in Tables 4.5 and 4.6. They correspond to a tire with similar characteristics.

a_0	1.3
a_1	-53.31
a_2	1190
a_3	588.6
a_4	2.5212
a_5	0
a_6	-0.5178
a_7	1
a_8	0
a_9	0
a_{10}	0
a_{11}	0
a_{12}	0
a_{13}	0

Table 4.5: Experimental parameters to calculate F_x

b_0	1.65
b_1	0
b_2	1688
b_3	0
b_4	229
b_5	0
b_6	0
b_7	0
b_8	-10
b_9	0
b_{10}	0
b_{11}	0
b_{12}	0
b_{13}	0

Table 4.6: Experimental parameters to calculate F_x

The next step was the implementation of Equation (4.5), using the given experimental parameters and following the Equations from (4.6) to (4.11), we got the results described in Figures 4.28 and 4.29.

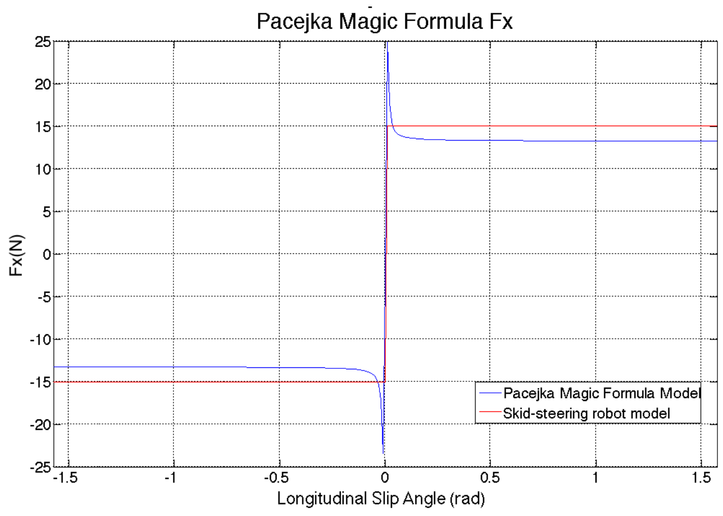


Figure 4.28: Pacejka magic implementation for F_x.

The results allow us to assess some similarities between the two models from the theoretical point of view. In the linear one, the forces acting in both longitudinal and transversal direction are constant for all angles, using

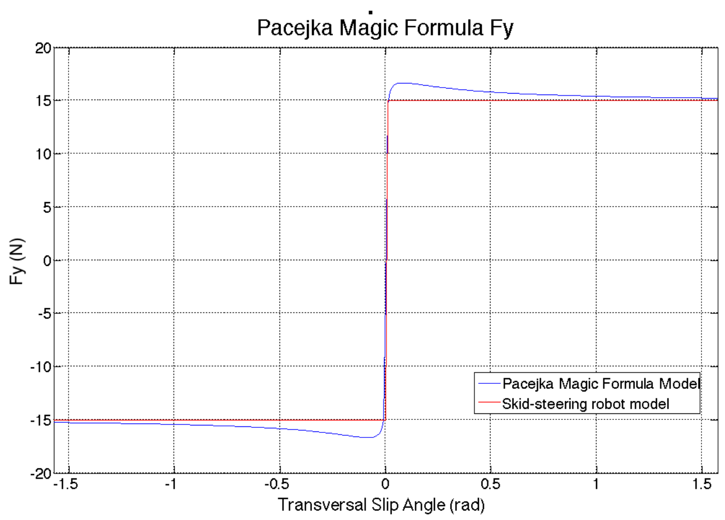


Figure 4.29: Pacejka magic implementation for F_y.

the Pacejka approximation we identify that for small angles the forces acting into the vehicle are higher for the same slip angle range. The most important results is that the magnitudes of the forces are comparable.

4.4.2 Simulink Comparison

For the comparison, we considered the same trajectories that were simulated in Sections 4.3.1 and 4.3.2, the variables that we will assess are:

- v_x: linear velocity.
- $\dot{\psi}$: angular velocity.
- Trajectory tracking comparison.

Through these results, we can conclude whether the robot has similar dynamics or not.

Comparison Trajectory 1

The input trajectory is the one illustrated in Figure 4.5. Figure 4.30 shows the comparison of these two variables in both models.

This result verifies the initial assumptions, since the variations for small angles are low. We did not expect to notice a notable difference because as we assumed in Section 2.2.2, the longitudinal slip angle is very small and in the linear approximation we neglected it, However, it was interesting to

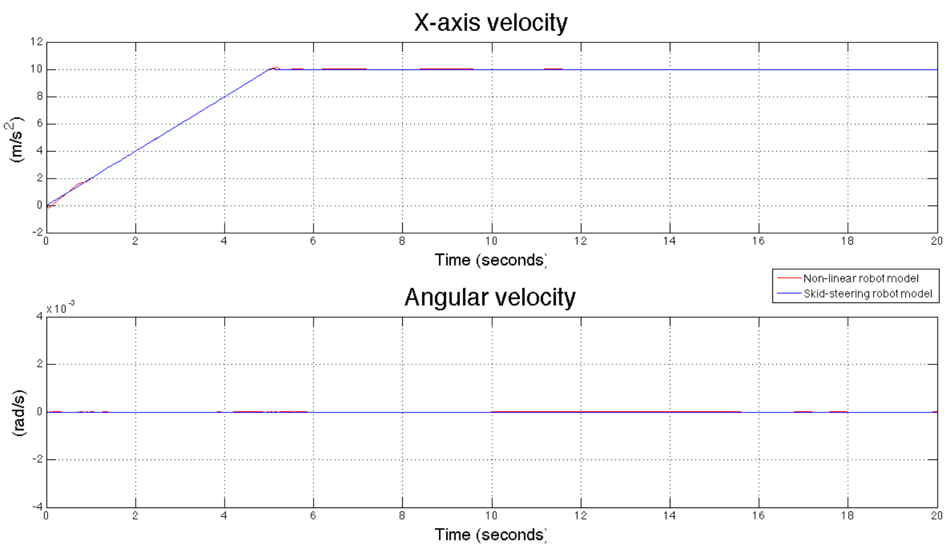


Figure 4.30: Comparative between models (kinematics).

analyze the behavior of the model when subjected to maximum velocity and maximum acceleration. Figure 4.30 verifies the model for these variables, since the difference between the plots is negligible. The only difference appears in the transients where the error increases.

Furthermore, the accuracy of results in Figure 4.30 allows us to anticipate that accuracy in the second comparison for the global variables is going to be high. The procedure was carried out and the results illustrated in Figure 4.31.

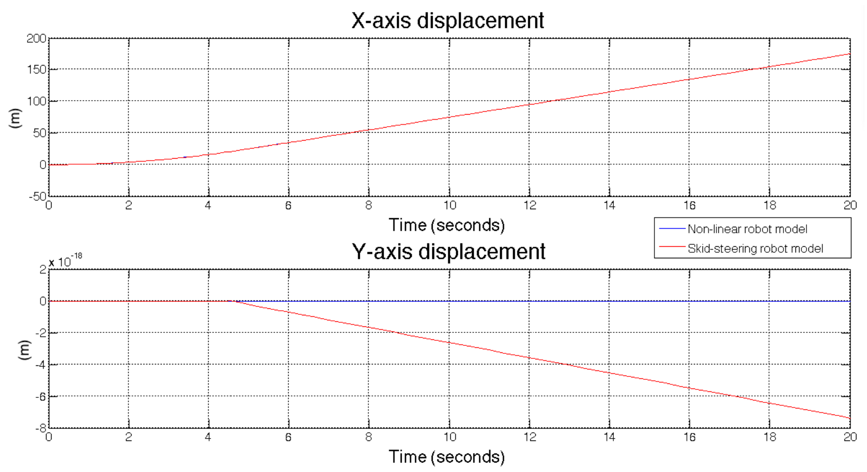


Figure 4.31: Comparative between models (tracking).

As expected, there is no major difference between the two models, since the dynamics modeling is really similar. All the longitudinal quantities in both models are following the same trend. It is true that the model that implements the Pacejka magic formula has more considerations, but none of these affect the vehicle in a trajectory like one evaluated in this section.

Comparison Trajectory 2

The input trajectory is the one illustrated in Figure 4.16, but in this case the robot faces only one curve without slope.

Unlike the previous experiment in which the forces were equivalent in both models, we will compare the resistive forces because the Pacejka implementation has a higher influence on these parameters for this test, Figure 4.32 shows the results obtained. We identify a higher variation of the system with respect to the non-linear version, however, this remains within an acceptable range. We note that in effect, all forces vary depending on the angle at which the vehicle is subjected, there exists a range of values in which the results follow the same trend, but one in which the models have a difference.

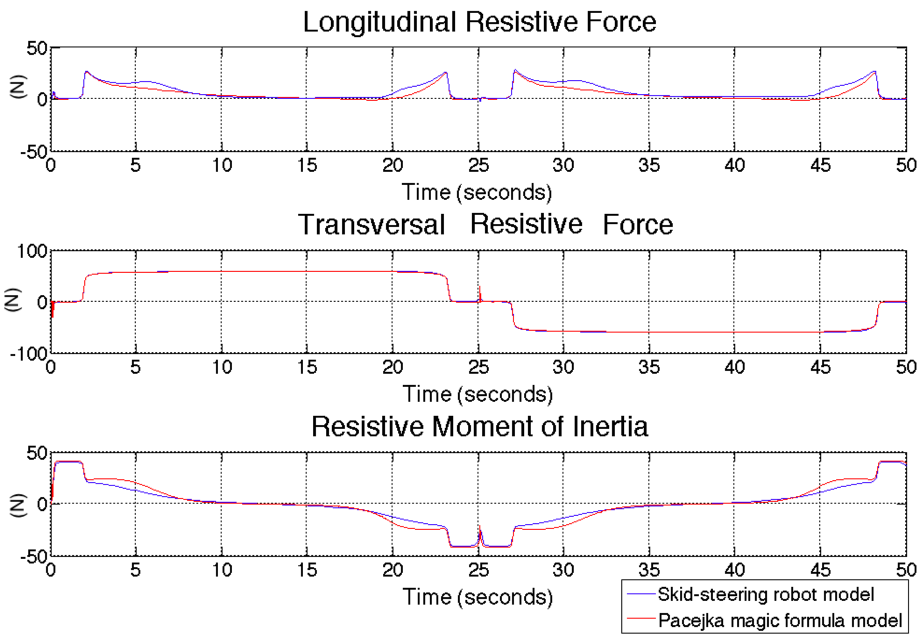


Figure 4.32: Comparative between models (resistive forces).

Figure 4.33 shows the tracking results for both models. The Pacejka approximation illustrates a more realistic path followed by the robot, since in this case real forces for small angles have been considered. Despite this, the results have not a high absolute error, which allows the final verification of the model of a skid-steering robot.

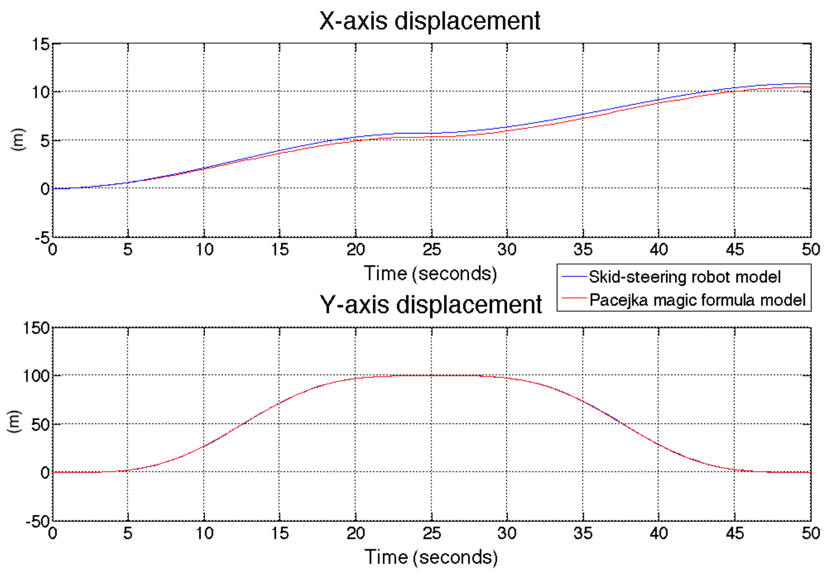


Figure 4.33: Comparative between models (tracking).

Chapter 5

Implementation of the Motor Control System

5.1 Introduction

In this chapter the structure of the mobile robot and its main components will be described. A more complex modeling of the DC-brushless drive than the one presented on chapter 2.3.1 will be introduced to explain the implementation of the motor control algorithm, the method is called *Field Oriented Control*. Several libraries written in C will be described to handle the sensors and the control, the codes are extended in Appendix A. In addition, an electronic board with a 32-bit microprocessor will be used to test the control system as a whole. However, due to the absence of the high power module, a smaller motor will be used for testing, instead of the vehicle's ones.

5.2 Vehicle Structure and Components

5.2.1 Chassis

The mobile robot mainframe can be seen in Figure 5.1. It is composed of a set of extruded aluminum beams which hold together all its components. The dimensions can be seen on table 5.1. It was designed in such a way to be flexible, allowing it to be expandable to more features like a robotic arm, a camera system or proximity sensors.

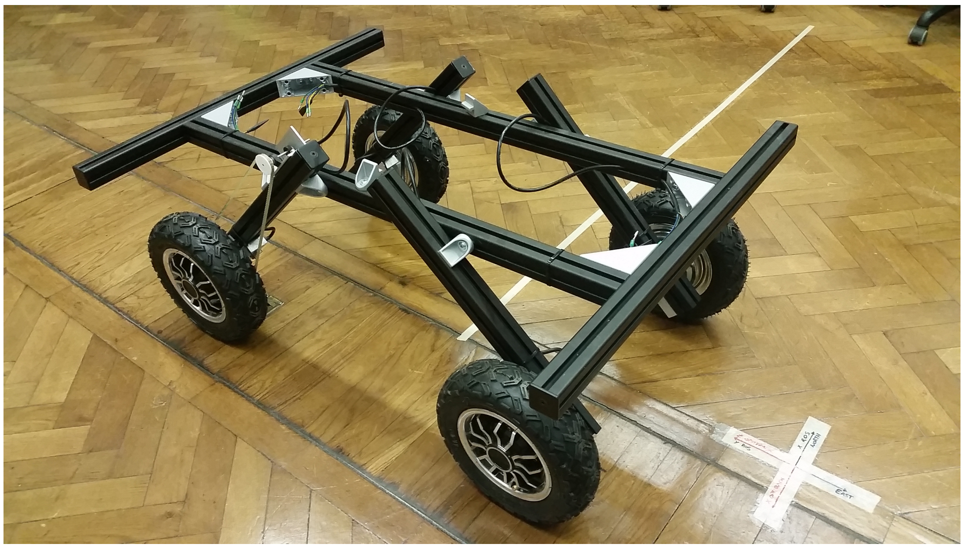


Figure 5.1: Chassis of the skid-steering robot.

Parameter	Value	Units
Height	0.485	m
Width	0.671	m
Lenght	0.75	m
Chassis material	Aluminium (Al)	-
Projected total mass	60	Kg
Max. total power	2000	W

Table 5.1: Main robot characteristics.

5.2.2 Batteries

The vehicle is powered by a set of three 12V lead-acid sealed batteries, seen in Figure 5.2, with a total capacity of 81 Ah. Although its low power-to-weight ratio, they are one of the most common and cheap rechargeable batteries on the market and are still used in many applications. They also have the advantage of being of the sealed type, allowing them to work in any position. Their specification are in Table 5.2. At a maximum speed of $10m/s$ in flat terrain, the vehicle will have a theoretical autonomy of 80 minutes with this configuration.

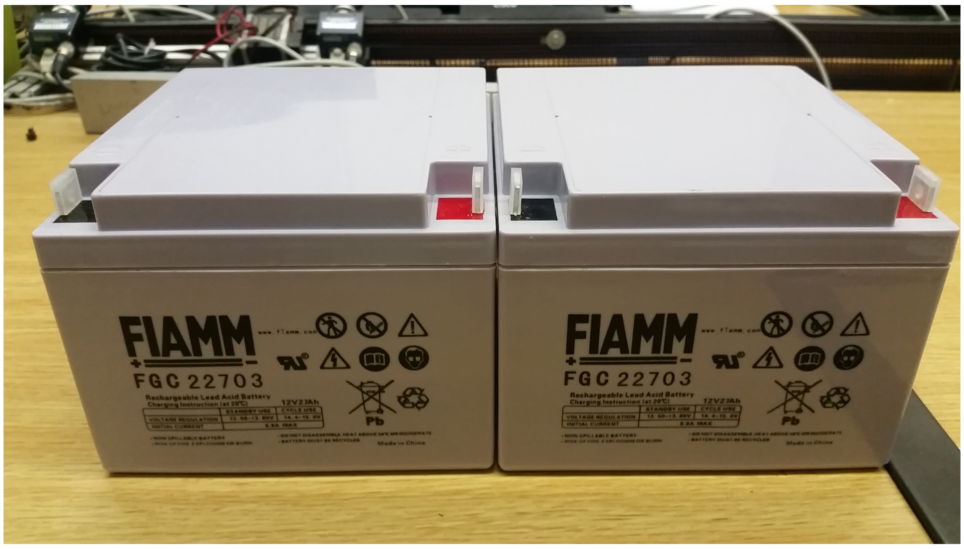


Figure 5.2: FIAMM FGC 22703 rechargable lead acid batteries.

Parameter	Value	Units
Voltage regulation	13.5 - 13.8	V
Initial current	6.8	A
Nominal Voltage	12	V
Nominal Capacty @ 20hr	27	Ah
Material	Lead (Pb)	-
Length	166	mm
Width	175	mm
Height	125	mm
Weight	8.5	Kg

Table 5.2: FIAMM FGC 22703 Specification.

5.2.3 Wheels and Motors

The electrical drives used in this system are the *HUB10GL* and a picture of them can be seen in Figure 5.4. In this solution, the wheels are coupled to the motor's axle, becoming unnecessary the need for a power transmission. The specifications can be seen in Table 4.4 and its dimensions in Figure 5.3. The maximum nominal power of each one is $500W$ which, according to our simulations, is more than enough to drive the system in the steepest slope of $45°$. The wheels attached to them have a radius of 12,7 cm, they are made of synthetic rubber and are filled with air.

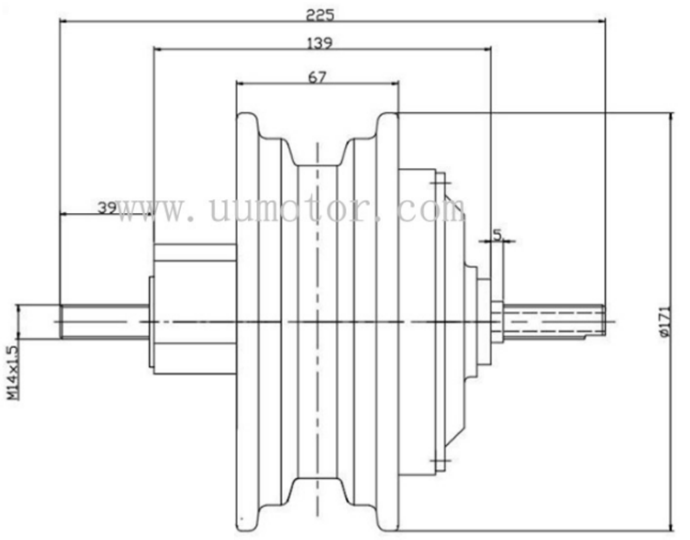


Figure 5.3: HUB10GL motors dimensions.

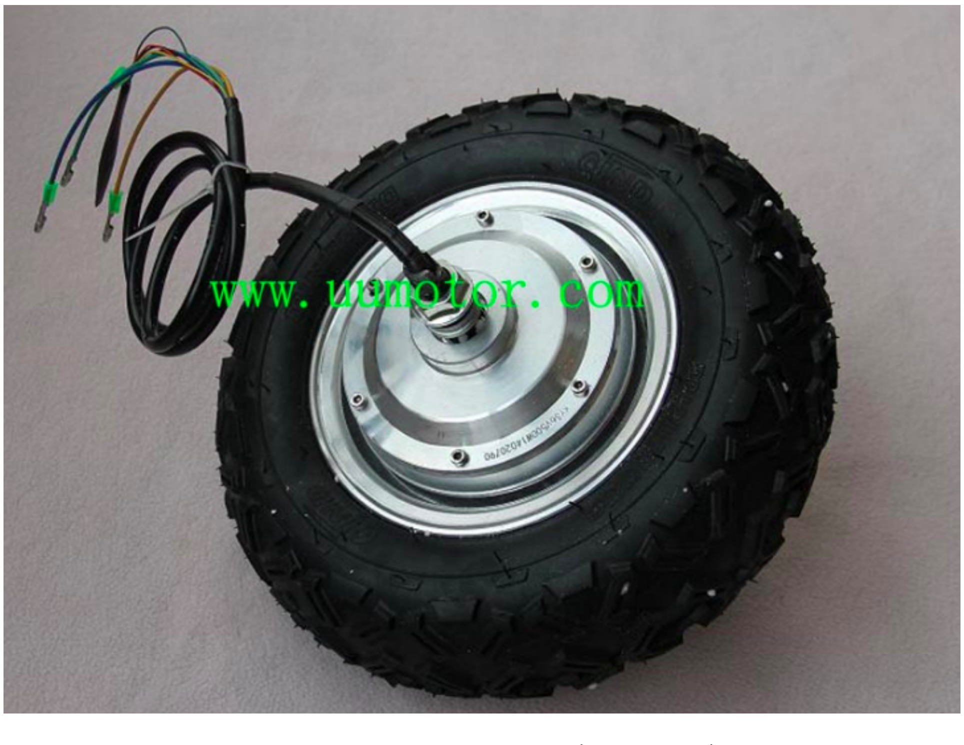


Figure 5.4: HUB10GL motor (real picture).

5.2.4 Position Sensors

Orbis Absolute Rotary Encoder

As we will see further in section 5.3, it is necessary to measure the rotor's location and velocity at all times. The typical approach to measure the position and its derivatives is by means of a rotary encoder attached to it. Which is an electromechanical device that transforms the shaft's angular position into a digital signal. Usually, the most cheap rotary encoders are the incremental encoders, which measures the amount of movement of the shaft and by integration or other means we obtain the position.

However, in this system we have decided to use the *Orbis^{TM} True Absolute Rotary Encoder* (Figure 5.5) which works by measuring the magnetic field of a magnet cylinder attached to the shaft. These component has the following advantages:

- It assigns a unique value to each position of the axle. i.e., there is no need to move the rotor to an initial known position to start the system.
- The magnetic and electronic parts are completed decoupled mechanically from each other, giving us more flexibility to decide where to install it.
- Is a digital system so there is no need for an analog-to-digital conversion, all the communications are by SSI or SPI protocol.

More information about this component can be seen in Appendix C.

5.2.5 Electronic Board

Each motor will be control with an electronic board from the brand Renesas called *YROTATE-IT-R5F523T5 Motor Control Kit*, the complete kit is shown in Figure 5.6. Its main characteristics are resumed in Table 5.3 and can be extended in Appendix D.

The hardware is based on the chip *R5F523T5*, which was developed by Renesas specifically for motor control solutions.

This solution was chosen because it comes already with several libraries for motor control and can be connected with a high power module to drive motors above 100W.

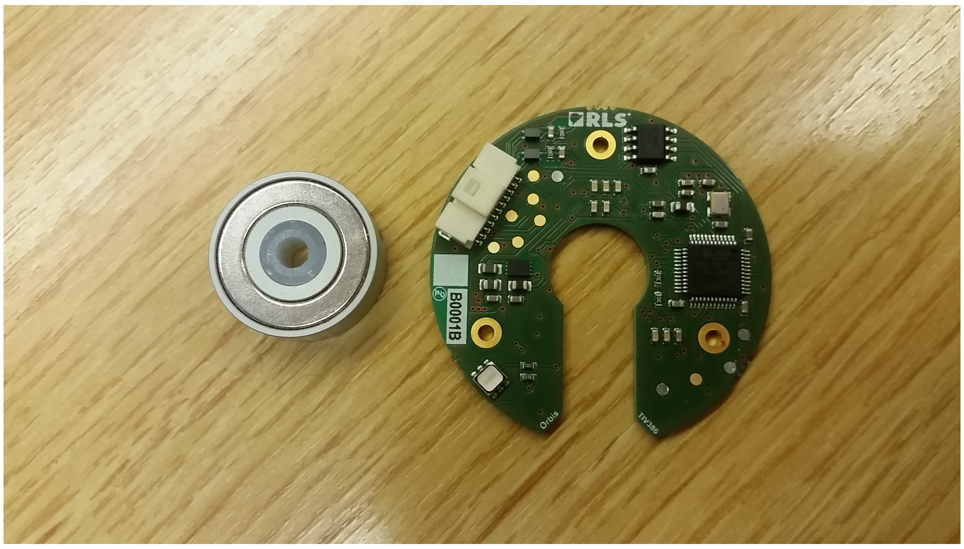


Figure 5.5: Orbis^{TM} true absolute rotary encoder.

Item	Specifications
Type of motors supported	3-phase PMSM, PMAC,BLAC, BLDC
Transistors used	Renesas Mosfets: RJK0654DPB, 60V, 30A
Power supply	up to 48VDC
Current detection	One or three shunts configuration ($10m\Omega$)
Microcontroller	RX23T (R5F523T5ADFM)
	64-pin LFQFP, 40MHz,
	64KB Flash, 10KB RAM
MCU performance	40MHz, 80DMIPs, 166 CoreMark
Key features	Floating Point Unit
	3-phase inverter Timer
	12-bit A/D Converter
Switching frequency	4KHz to 64KHz
	16KHz by default (PWM frequency)
Control loop frequency	4KHz to 16KHz, 8KHz by default
Tool used	e^2 studio 4.0.2.008.
	RXC Toolchain
	CC RX version v2.03.00

Table 5.3: Main characteristics of the Develoment board.

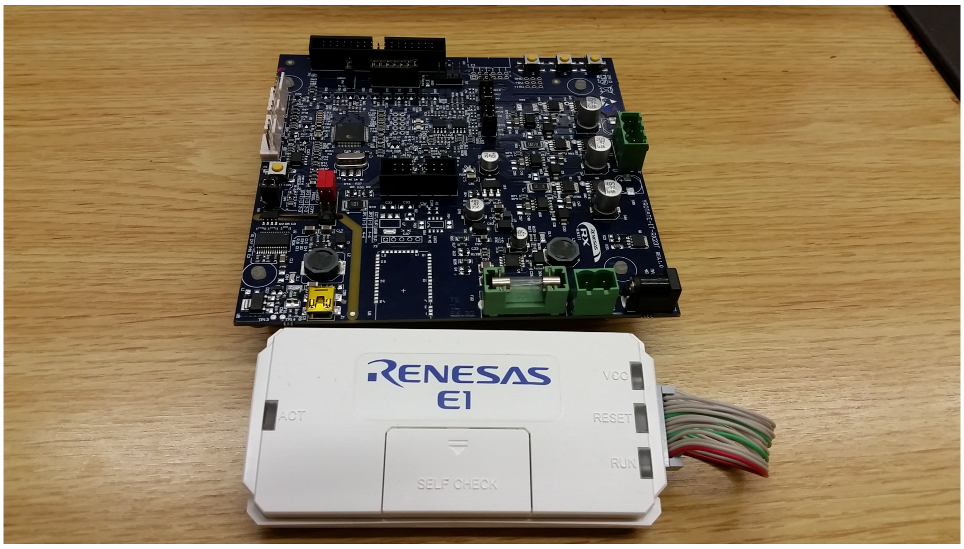


Figure 5.6: YROTATE-IT-R5F523T5 Motor Control Kit from the brand Renesas.

5.3 Motor Control Algorithm

In Section 2.3.1 we derived a simplified DC brushless motor model that retains the most important dynamic and mechanical characteristics, this model was good enough to perform tests on the motor and conclude about its performance. However, for control purposes, we require a more complex model with a higher accuracy. Here, we introduce a model that will allow us to implement more advanced control techniques.

5.3.1 Field Oriented Control

The control method to be implemented in the board is called Field Oriented Control. The idea is to recreate the same situation of a DC drive, in which, one current defines the torque and the other the electromagnetic flux.

In a DC brushless, the current flowing in each winding produces a magnetic field vector. For any position of the rotor, there is an specific field vector direction that maximizes the torque. For example, if the permanent magnet of the rotor is aligned with the stator field, then there is no movement, but a force acting to compress the bearings of the motor, while if there is 90°displacement between them, the torque produced will be maximum [8]. Field Oriented Control is a technique in which, by means of some mathematical transformation, we can convert the stator's currents in a direct and quadrature current equivalences, as can be seen in Figure 5.7. The direct current i_d does not produce torque while the quadrature i_q is responsible for the torque of the electrical drive.

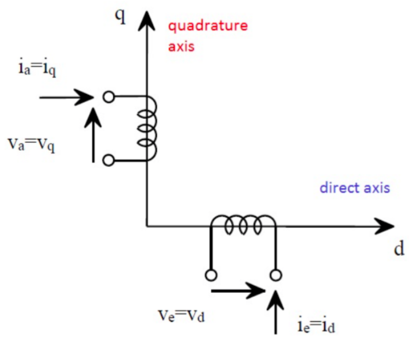


Figure 5.7: Equivalent model of the fictitious machine for FOC control.

This equivalent machine is based on the Park and Clarke transformations. Which will be further discussed.

Park and Clarke Transformations

A three-phase machines can be described by their electrical equations. The modeling is usually complex due to the fact that all flux linkages, induced voltages, and currents are changing continuously, as the circuit is in relative motion. This complex analysis is solved by the use of Clarke and Park transformations, the aim of this method is to decouple variables and to solve equations involving time varying quantities, to do that, the the method refers all variables to a common reference frame.

Clarke transformation

Clarke transformation allows us to convert three-phase quantities (I_a, I_b, I_c) into equivalent two-phase direct/quadrature quantities in a stationary reference frame named I_α and I_β:

$$I_\alpha = \frac{2}{3}I_a - \frac{1}{3}(I_b - I_c) \qquad I_\beta = \frac{2}{\sqrt{3}}(I_b - I_c) \tag{5.1}$$

When the currents are balanced, that is when $I_a + I_b + I_c = 0$, I_a, I_b, and I_c can be transformed to I_α and I_β as:

$$I_\alpha = I_a \qquad I_\beta = \frac{1}{\sqrt{3}}(2I_b + I_a) \tag{5.2}$$

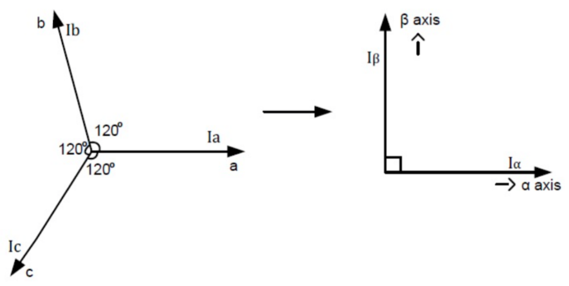


Figure 5.8: Clarke transformation.

Park transformation

By using Park transformation the two-axis orthogonal stationary reference frame quantities I_α and I_β are transformed into rotating reference frame quantities I_d and I_q.

$$I_d = I_\alpha \cos\theta + I_\beta \sin\theta \qquad I_q = I_\beta \cos\theta - I_\alpha \sin\theta \tag{5.3}$$

where θ is the rotation angle.

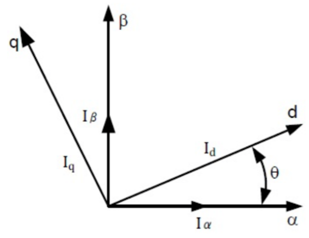


Figure 5.9: Park transformation.

Inverse Clarke Transformation

The transformation from a two-axis orthogonal stationary reference frame V_α and V_β to a three-phase stationary reference frame V_a, V_b, V_c is performed by using inverse Clarke transformation as follows:

$$V_a = V_\alpha \tag{5.4}$$

$$V_b = \frac{-V_\alpha + \sqrt{3}V_\beta}{2} \tag{5.5}$$

$$V_c = \frac{-V_\alpha - \sqrt{3}V_\beta}{2} \tag{5.6}$$

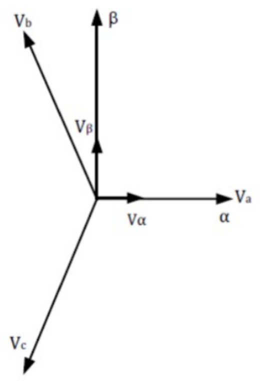


Figure 5.10: Inverse Clarke transformation.

Inverse Park transformation

The quantities in rotating reference frame V_d and V_q are converted into two-axis orthogonal stationary reference frame V_α, V_β by using inverse Park transformation as follows:

$$V_\alpha = V_d \cos(\theta) - V_q \sin(\theta) \qquad V_\beta = V_q \cos(\theta) + V_d \sin(\theta) \tag{5.7}$$

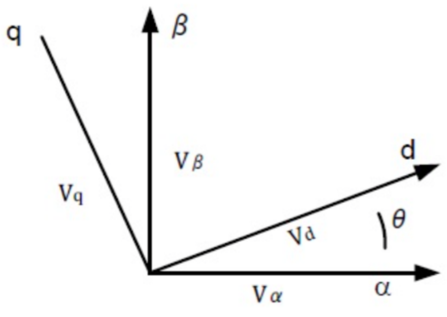


Figure 5.11: Inverse Park transformation.

DC Brushless Motor Model

Once we have implemented the proper transformation from three-phase quantities to rotating reference frame quantities, we can proceed to design the motor model. The electrical and mechanical parts can be written in the form of a second-order state-space model [11].

$$\frac{d}{dt}i_d = \frac{1}{L_d}v_d - \frac{R}{L_d}i_d + \frac{L_q}{L_d}p\omega_m i_q \tag{5.8}$$

$$\frac{d}{dt}i_q = \frac{1}{L_q}v_q - \frac{R}{L_q}i_q - \frac{L_d}{L_q}p\omega_m i_d - \frac{\varphi_{0PM}}{L_q}\omega_m \tag{5.9}$$

$$\tau_e = p\left[\varphi_{PM}i_q + (L_d - L_q)i_d i_q\right] \tag{5.10}$$

$$\frac{d}{dt}\theta_m = \omega_m \tag{5.11}$$

$$\frac{d}{dt}\omega_m = \frac{\tau_e - \tau_r - B\omega_m}{J} \tag{5.12}$$

where these equations are represented in rotor reference frame (d-q frame). All quantities in the rotor reference frame are referred to the stator. Table (5.4) shows the meaning of each variable implemented in the model.

Symbol	Meaning	Measurement unit
J	Combined inertia of rotor and load	[Kg·m^2]
B	Combined viscous friction of rotor and load	[N]
τ_r	Shaft static friction torque	[Nm]
ω_m	Angular velocity of the rotor (mechanical speed)	[Rad/sec]
L_q, L_d	d and q axis inductance	[Henry]
R	Resistance (phase to phase) of the stator windings	[Ohm]
p	pole pairs	real number
T_e	Electromagnetic torque	[Nm]
φ_{PM}	Flux induced by the permanent magnets	[Weber]

Table 5.4: Meaning of variables required to define de motor model.

The equations to be implemented are:

$$\begin{cases} i_d = \frac{1}{sL_d+R}(v_d + L_q p\omega_m i_q) \\ i_q = \frac{1}{sL_q+R}(v_q - L_q p\omega_m i_d - \varphi_{PM} p\omega_m) \end{cases} \tag{5.13}$$

From Equations (5.10) and (5.13) we can see a direct relationship between the current i_q and the electromagnetic torque produced by the machine. A control strategy will be to control currents i_d and i_q: The first one must remain as close to zero as possible to minimize its effects onto the quadrature current and the second one to generate the desired τ_e.

The Figure 5.12 shows the block diagram of a motor control using FOC, where θ denotes the mechanical angle of the shaft. For this scheme, the minimum mandatory measurements must be the phase currents and the motor mechanical speed from which we can obtain the mechanical angle.

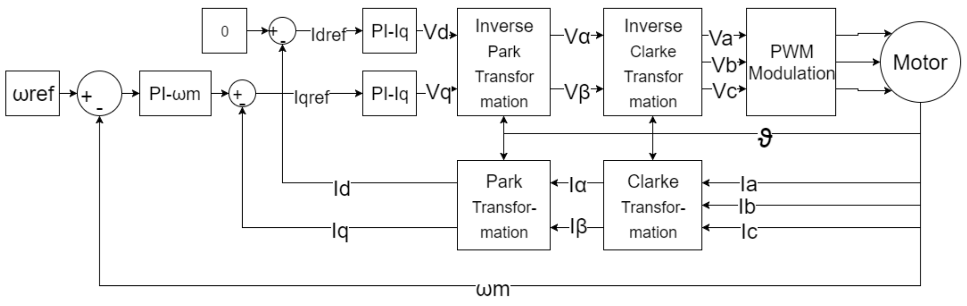


Figure 5.12: Field Oriented Control block diagram for a Brushless DC motor.

5.4 Experimental Setup

An experimental setup to test the motor control system into the electronic board will be introduced. The purpose of the test will be to drive a small DC brushless motor and study the behavior of the electronic board and the close loop system response.

This motor for test comes directly coupled with an incremental encoder that will be used to get the rotor position and velocity for control, while another velocity measurement will be obtained using the position encoder described on section 5.2.4, both data will be compared to conclude about the different sensors. Several libraries for the implementation were necessary, for more details about them and the rotary sensor used can be consulted on Appendix A.

5.4.1 Motor Description

The motor test used is the 'MSSI-040H-027' from the brand Wittenstein Cyber Motor, it is shown in Figure 5.14. The shaft is directly coupled to an encoder with 2000 segments per revolution, giving us a resolution of 0.18°. Specifications appears in the Table 5.5 and can be extended in Appendix B. The controller design used to tune this system is the one described in Section 3.

Additionally, the motor has an inertial load of 15 gcm^2, so the total mass moment of inertia is 38.5 gcm^2. It is supplied by 24V by from a series of two of the 12V batteries used to power the vehicle. Figure 5.13 shows the final setup of the experiment.

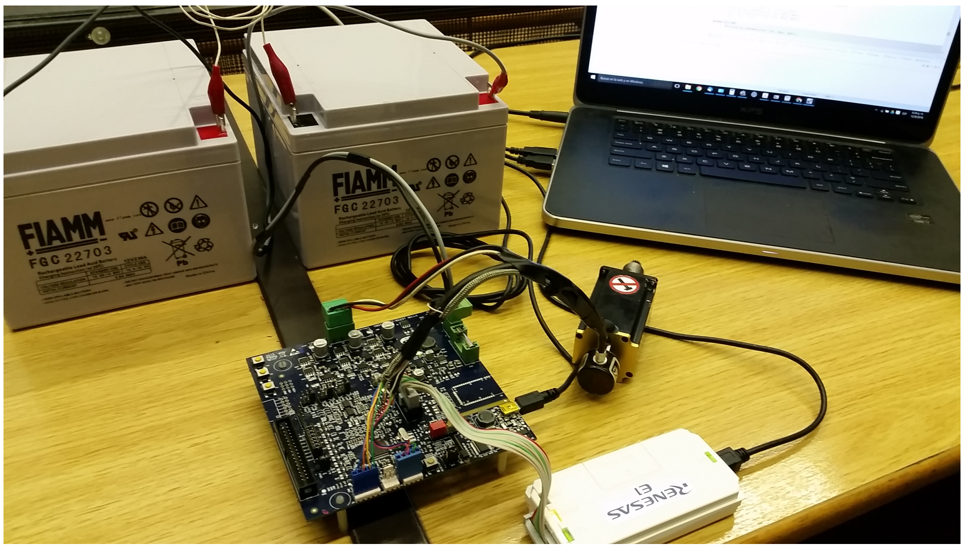


Figure 5.13: 'Final setup of the experiments.

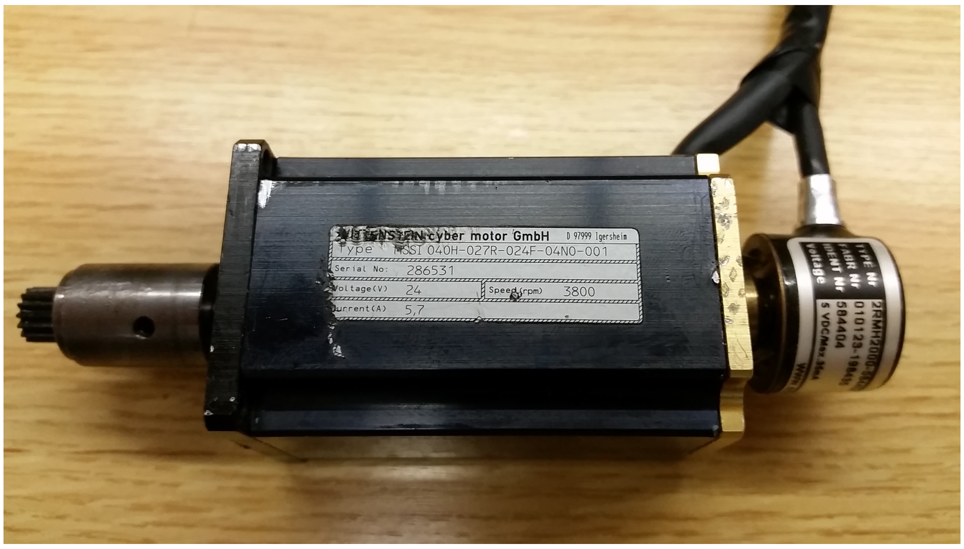

WITTENSTEIN cyber motor GmbH D 97999 Igersheim
Type MSSI 040H-027R-024F-04N0-001
Serial No: 286531
Voltage(V) 24
Speed (rpm) 3800
Current(A) 5,7

Figure 5.14: 'MSSI-040H-027' motor from the brand Wittenstein Cyber Motor.

Parameter	Value	Units
DC Bus Voltage	24	V
Max. Torque	0.724	Nm
Max. Current	16	A
Continuous Stall Torque	0.18	Nm
Continuous Current	3.9	A
No-Load Speed	6200	rpm
Nominal Torque	0.163	Nm
Nominal Current	3.7	A
Nominal Speed	5100	rpm
Torque Constant	3.8	Ncm/A
Winding Temperature	140	°C
Termal Resistance	3.25	K/W
Terminal Resistance	0.6	ohms
Terminal Inductance	0.4	mH
Eletrical Time Constant	0.7	ms
Mass Moment of Inertia	23.5	gcm^2
Number of phases	3	-
Weight of the Motor	340	g

Table 5.5: MSSI-040H-027 motor data specifications.

5.5 Control Loops Experiments

5.5.1 Current Control Loop Validation

In order to validate the current control loop we will propose a test, afterward we will explain the reasons why we consider this procedure appropriate to validate the loop and finally, we will present the expected results.

Test 1: Step Response

The test is the most basic and involves making a step in order to measure the response parameters. With this experiment, we expect to compare the results obtained in MATLAB (the absolute error will provide information on design accuracy). The most important results for this test will be [4]:

- Loop gain k_i.
- Rise time T_r.

- Time constant T.

First, Figure 5.15 shows the comparison between simulations in MATLAB and experiments in the real motor. Similarly, Table 5.6 contains the values with the results and the corresponding error.

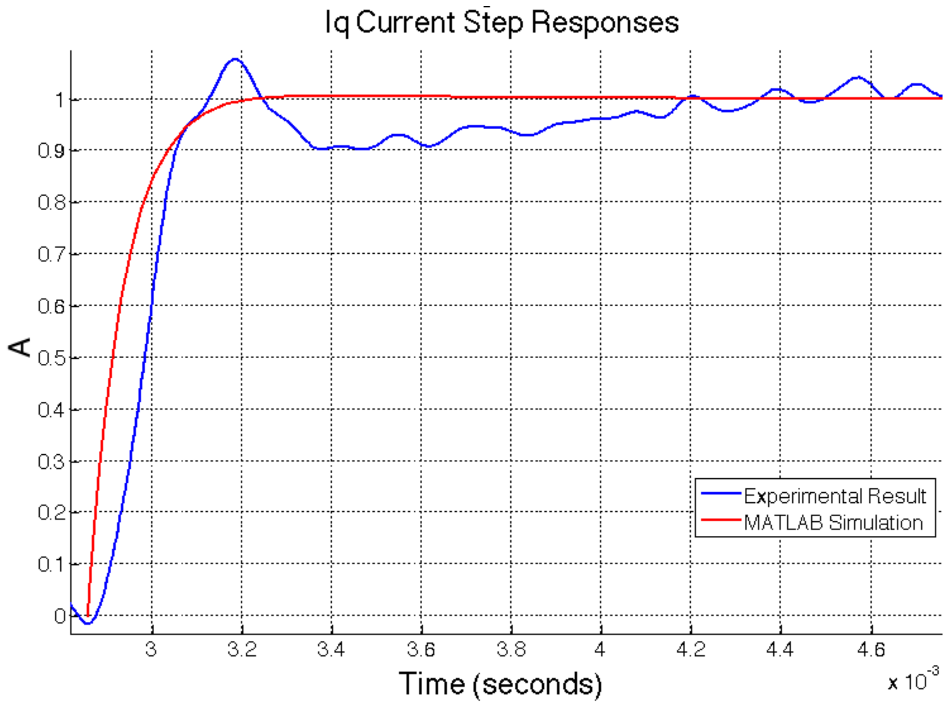


Figure 5.15: Step response comparison (current control loop).

Parameter	Simulated Value	Real Value	Units	Absolute Error
Rise Time	0.19	0.17	ms	0.02 ms
Settling Time	0.62	1.04	ms	0.42 ms
SettlingMin	0.9	0.905	-	0.005
SettlingMax	1	1.08	-	0.08
Overshoot	-	8.02	%	-
Peak	1.01	1.08	A	0.07A
Peak Time	0.95	0.32	ms	0.63 ms

Table 5.6: Step response comparison for validation (current loop).

Analyzing this information, we can see that the responses are correlated. The absolute error for the rise time was 0.02 ms, which is almost negligible, while for the time constant was 0.084 ms. The most relevant result is the small absolute error, since it allows us to verify that the controllers have a similar behavior.

Although the set point is not accurate, this is not necessary a bad result, as we have to take into consideration that we assumed for the modeling a very simply electrical machine and many non linearities were ignored like the power module switching effect, the measurement errors in the sensors, etc.

Another source of error could be the motor angular position, at the beginning of the test, the rotor was aligned with the first phase and physically fixed, however, any minor movement could have made a mismatch between the theoretical position and the real one.

In any case, the results are considered satisfactory as a first validation of the designed controller, the time response and gain values are consistent with the results expected by simulations in MATLAB.

5.5.2 Speed Control Loop Validation

Test 1: Step Response

The test consists in the imposition of a constant speed reference to evaluate the speed control loop. The aim is to assess the parameters of the transient response, and then, proceed to compare with the MATLAB results. Figure 5.16 shows the results for a comparison between Simulink tests and implementation on the Renesas board. On the other hand, Table 5.7 shows the numerical results and the respective errors for the parameters evaluated.

Parameter	Simulated Value	Real Value	Units	Absolute Error
Rise Time	0.04	0.024	s	16 ms
Settling Time	0.22	0.17	s	0.05 s
SettlingMin	0.9	0.9	-	-
SettlingMax	1.19	1.51	-	0.32
Overshoot	19.87	51	%	31.13 %
Peak	1.191	1.51	-	0.32
Peak Time	0.1	0.058	s	42 ms

Table 5.7: Step response comparison for validation (speed).

The results allow to verify the similarity between the responses. The

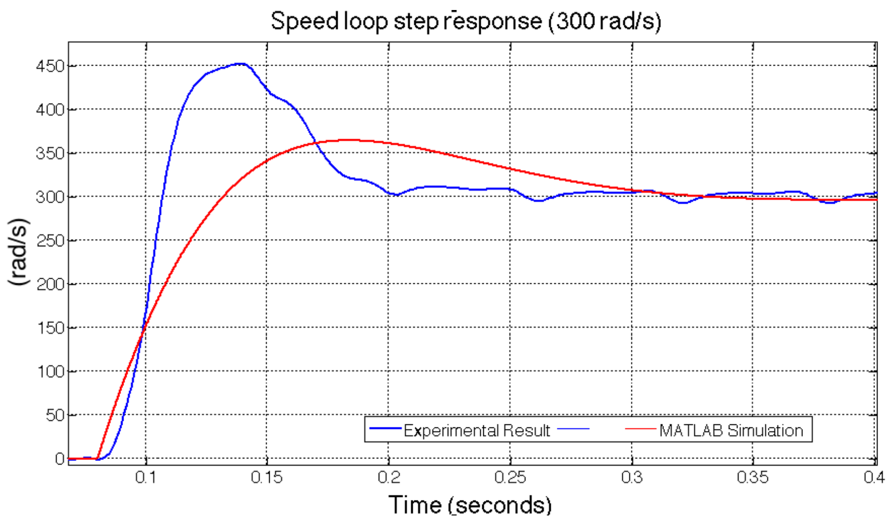


Figure 5.16: Step response comparison (speed control loop).

rise time has an absolute error of 16 ms, which is acceptable for the speed loop. Moreover, the settling time has a difference of 0.05 seconds with respect to the real one. The peak in the simulation has a magnitude of 19 % with respect to the settling point, while the test has a magnitude of 51 %. The data obtained verify that the speed controller has an acceptable performance. Additionally, it checks that the current control loop can be taken as a unitary gain that does not affect the design of the speed control loop.

We could get a better response from the system, however, the results obtained allow to proceed to the next tests which consist in the realization of experiments on the real motors implementing this control methodology.

Test 2: Ramp Response

The second test consists in the imposition of a ramp speed reference. The aim is to assess the steady state error, and then, compare with the MATLAB simulations. Figure 5.17 shows a comparison between simulations and experiments on the Renesas board, the ramp reference is illustrated as well. We can see in the grapgh that the experimental speed oscillates around the reference, this is because of errors in the test, since the magnetic sensor is not perfectly aligned.

Regarding the steady state error, in the simulations, it takes 0.18 seconds to stabilize, while in the experiments the value was 0.08 seconds. This is not a notable difference, because this parameter during the simulation stabilizes asymptotically. On the other hand, the maximum error is 6.15 rad/s in the simulation and 9.68 rad/s in the experiments, then the behavior is still similar even in this comparison.

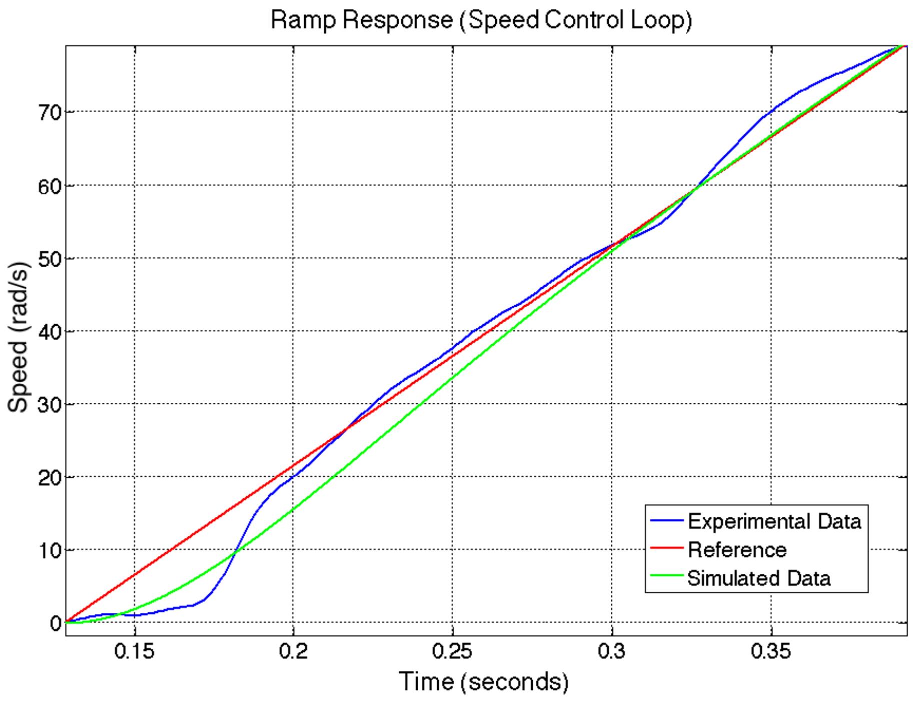


Figure 5.17: Ramp response comparison (speed control loop).

Finally, the most important result is that both experiments and simulations verified that these signals follow the reference without steady state error. As the PI controllers are designed to have a reaction of this type, we can consider this a successful test. In conclusion, if we compare the absolute errors of each of the above parameters, we can reach a second validation for the speed control loop.

Chapter 6

Conclusions

The main objective of this thesis is to control the in-wheel motors for an outdoor skid-steering robot. We could focus the analysis from different points of view, for instance, Simulink experiments allow not only to compare the motors modeling but also the dynamic model implemented. Similarly, analyzing the Pacejka model we could verify the mechanical modeling of the vehicle and the peformance of the control respecto to the new dynamics. First, from the electrical point of view we can derive the following conclusions:

- If we are interested in the robot to face steep slopes (more than 30 degrees), it is advisable to perform tests in the maximum speed for which the vehicle can climb without exceeding the limits of nominal current and torque as shown in Figures 4.19 and 4.17. During the experiments in Section 4.3.2, we identified the motors working above the nominal values. Although it is possible to move with these parameters, the time during which the motor is able to do so is unknown and is related to the maximum nominal and peak currents that the motor can manage due to the increase of temperature .

- In order to accomplish a better control loop tuning, we need to characterize the motor to get the real parameters describing its dynamics, for instance, resistance, inductance and maximum currents. In the simulations, the motor is able to run with any current below the stall value, but in reality that is not possible because it can overheat, so we recommend a set of experiments to carry out a motor identification not only to know the limit values, but also to optimize the designed control loop.

- For a trajectory as described in Figure 4.21, we could identify that the motor has a consumption of about 1.75 Ah/min, with a battery capability of 40.5 Ah, the motors theoretically could move for a time between 23 and 25 minutes without being recharged. This result allows to have a first sight of the autonomy of the robot. This is the worst scenario, therefore we expect an increased runtime on more realistic trails.

- The trajectory generator developed in Section 2.4 proved that this implementation may optimize motors performance. By modifying the timing law, it is possible to control the maximum current values reached in critical points. For instance, for a steep slope it would be possible to reduce the speed at a certain point, and after that, increase again to maximum speed. The fact that the generator gives the possibility to control these variables will be an advantage in a future implementation.

On the other hand, analyzing the implementation of current and speed control loops, we can derive the following results:

- In Sections 3 and 5 we implemented the same control methodologies for current and speed loops, in both cases the results verified the theory. Simulink experiments led to the conclusion that the parameters of the transient response were satisfied, which allowed us to proceed to a validation process. When we carried out the experiments using the Renesas board, the results gave us a second verification on the designing accuracy. Once again, the system response had correlation with the transient response obtained in MATLAB.

- For the current control loop we found that in Figure 5.15, the loop gain meets the requirements. However, the most important result was to check the system time constant and rise time. These are the results that allowed to validate the design as they are in correspondence with the bandwidth verified using Simulink. In addition, the control system was able to run even with disturbances carried on the load, thus proving a verification on the robustness.

- For the speed control loop we were able to validate through the implementation in the Renesas board that the response was acceptable for variables such as the gain, the response time and the overshoot. Nevertheless, the most interesting result was that the shape follows the trend from the simulations and the steady state error is zero. The critically damped response predicted in the simulations was validated in experiments. Finally, we tested the step response for different speeds and in all cases, we got data consistent with the simulations, thus we can conclude that the controller fulfills the requirements.

Regarding the implementation of the nonlinear model using the Pacejka magic formula, it is important to clarify that the weighting parameters used in Tables 4.5 and 4.6 are not exactly those of the in-wheel motor described in Section 4.3. However, we saw in Figures 4.28 and 4.29 a comparison between the models, these graphics shown how the forces change depending on the steering angle in both models. We considered that the fit of these parameters is enough because the difference between the two graphs is small, in addition, simulations works similarly for both models.

One of the developments to be accomplished in the future is to perform a set of tests to get real Pacejka parameters. Thus, one could obtain more realistic results about what is happening with the vehicle when the wheels slip. Although this approach does not follow any physical model, it is accurate and simple enough when considering nonlinearities in the wheel, that is the reason why this model was our selection.

The similarity between the linear model and the nonlinear one is evident, in part this is because the dynamics in the two models is calculated using the same methodology, the biggest change was the selected coordinate system. In fact, the only difference in the equations is the contribution of the wheels, which considers all possible nonlinearities. The simulations were carried out with a friction coeficients $\mu_x = 0.1$, $\mu_y = 0.1$ and an instant center of rotation $ICR_x = 0.1$. Varying these values the margin of error between models increases, but these are typical values, then the results are still acceptable. Simulink tests allowed to check the linear model of the robot, it is not as accurate as the Pacejka implementation but it is enough to test the control loops and to monitor the tracking of trajectories.

Finally, it is necessary to emphasize that despite the success of the results in Simulink, it is not enough to completely verify the models. The characterization of all parts of the vehicle must be performed and a series of experiments must be accomplished to conclude about the real accuracy of the control loops. Once the real parameters of the motor are available, one can track trajectories and variables in real time to assess the performance of the whole system.

Bibliography

[1] K. J. AAström and T. Hägglund. *PID Controllers: Theory, Design, and Tuning.* Instrument Society of America, Research Triangle Park, NC, 2 edition, 1995.

[2] Nicoló Bachschmid, Stefano Bruni, Andrea Collina, Bruno Pizzigoni, Ferruccio Resta, and Alberto Zasso. *Fondamenti di meccanica teorica e applicata*, chapter 2. McGraw-Hill, 1st edition, 2003.

[3] Luca Caracciolo, Alessandro De Luca, and Stefano Iannitti. Trajectory tracking control of a four-wheel differentially driven mobile robot control for four-wheel independent driving electric vehicle. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, volume 4, pages 2632–2638, Marriott Hotel, Renaissance Center, Detroit, Michigan, May 1999. IEEE Robotics and Automation Society, IEEE.

[4] D. W. Clarke. Sensor, actuator and loop validation, Aug 1994.

[5] Reza N Jasar. *Vehicle Dynamics: Theory and Application*, chapter 3. Springer, 2nd edition, 2014.

[6] Reza N Jasar. *Vehicle Dynamics: Theory and Application*, chapter 3.5.1. Springer, 2nd edition, 2014.

[7] Reza N Jasar. *Vehicle Dynamics: Theory and Application.* Springer, 2nd edition, 2014.

[8] J. P. John, S. S. Kumar, and B. Jaya. Space vector modulation based field oriented control scheme for brushless dc motors. In *Emerging Trends in Electrical and Computer Technology (ICETECT), 2011 International Conference on*, pages 346–351, March 2011.

[9] Krzysztof Kozlowski and Dariusz Pazderski. Modeling and control of a 4-wheel skid-steering mobile robot. *International Journal of Applied Mathematics and Computer Science*, 14(4):477–496, 2004.

[10] Werner Leonhard. *Control of Electrical Drives.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.

[11] Kim Mun-Soo, Dall-Sup Song, Yong-Kil Lee, Tae-Hyun Won, Han-Woong Park, Yong-Il Jung, Man Hyung Lee, and H. Lee. A robust control of permanent magnet synchronous motor using load torque estimation. In *IEEE International Symposium on Industrial Electronics, ISIE 2001.*, volume 2, pages 1157–1162, Paradise Hotel, Pusan, Korea, June 2001. IEEE, IEEE.

[12] Katsuhiko Ogata. *Modern Control Engineering*, chapter 12. Aeeizh, 4th edition, 2002.

[13] Hans Pacejka. *Tyre and Vehicle Dynamics*, chapter 4.3. Elsevier, 2nd edition, 2006.

[14] Hans Pacejka, Egbert Bakker, and Lars Lidner. A new tire model with an application in vehicle dynamics studies. *SAE*, pages 2–13, April 1989.

[15] Sneha S. Patil, Gajanan M. Malwatkar, and Jayant V. Kulkarni. Performance enhancement in disturbance rejection by pid controller in series with low pass filter. In *2012 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, volume 1, pages 1–5, Tamilnadu College of Engineering Coimbatore-641659. Tamilnadu, India, Dec 2012. IEEE, IEEE.

[16] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*, chapter 11. Springer, 1st edition, 2009.

[17] Hongpeng Wang, Junjie Zhang, Jingang Yi, Dezhen Song, Suhada Jayasuriya, and Jingtai Liu. Modeling and motion stability analysis of skid-steered mobile robots. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pages 4112–4117, Kobe International Conference Center, Kobe, Japan, May 2009. IEEE.

[18] Jens Wittenburg. *Dynamics of Multibody Systems.* Springer, 2nd edition, 2007.

[19] Xiaodong Wu, Min Xu, and Lei Wang. Differential speed steering control for four-wheel independent driving electric vehicle. *International Journal of Materials, Mechanics and Manufacturing*, 1(4):355–359, November 2013.

[20] Jingang Yi, Hongpeng Wang, Junjie Zhang, Dezhen Song, Suhada Jayasuriya, and Jingtai Liu. Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation. *IEEE Transactions on Robotics*, 25(5):1087–1097, October 2009.

[21] Wei Yu, Oscar Ylaya, Emmanuel G., and Patrick Hollis. Analysis and experimental verification for dynamic modeling of a skid-steered wheeled vehicle. *IEEE Transactions on Robotics*, 26(2):340–353, April 2010.

Appendix A

Libraries and Sensors Used for the Test

A.1 Rotary Encoder

The output of this type of device is a couple of train pulses called A and B with a phase shift between them of 90 degrees. When the shaft rotates, the amount of pulses, which is related to the mechanical assembly, gives us the angular degree, while the phase gives us information about the sense in which is rotating. However, it is important to note that we have no information if the rotor is not moving (no pulses on the outputs) and in any case, it does not gives us the absolute angular position but the amount of movement of the shaft. This, in turn means that, in order to obtain an exact position, it is necessary to initialize the system to a known location from which start counting the pulses.

The *R5F523T5* has already built-in counters to make encoder measurements. In our development board, the encoder input A and B are connected to pins 40 and 41 respectively which are the inputs MTCLKA and MTCLKB of the microcontroller's counter MTU1.

The MTU1 is a 16-bit counter that increments or decrements the register MTU1.TCNT depending on the mode of operation. For our purpose, we will use 'Phase Counting Mode 1', whose behavior is described on Figure A.1. On each edge, from any of the signals A o B, the counter TCNT is incremented or decremented depending on which signal leads, i.e. which one comes first.

It is easy to see from Figure A.1 that each cycle increments or decrements MTU1.TCNT 4 times. Then, we can calculate the angular position using the following equation:

$$angle[radians] = \frac{2\pi \times n}{4 \times n_{pul\_rev}} \tag{A.1}$$

where n is the number of pulses counted and n_{pul_rev} is the number of pulses per revolution, a characteristical parameter of the encoder.

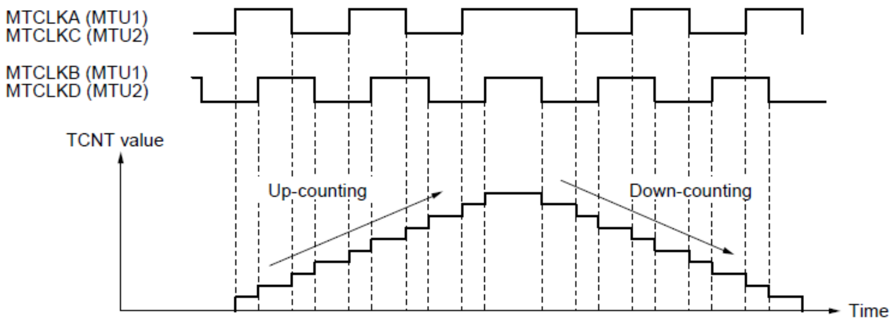


Figure A.1: Graphical description of MTU1 on 'Phase Counting Mode 1' operating mode. Courtesy of Renesas.

Due to the limited number of pulses in each sampling period, the shaft position measurement is corrupted by a quantization noise, therefore, a speed measure will result into a rather noisy value which is dependent on n_{pul_rev}. Bigger values implies lower noisy measurements, but it would increase the monetary cost of the sensor. In order to overcome this problem, it is necessary to filter the speed. The algorithm implements a moving average filter but it can be updated to more complex filters if necessary.

A new library was implemented to read the values from the encoder based on the original one given by Renesas. It contains the following functions:

- **Encoder_Init(void):** configures the ports and the counter MTU1. Must be called at the beginning of the configuring part of the main loop.
- **Encoder_Read(void):** updates the value of the mechanical angle and the mechanical speed. It must be called at every sample time.
- **Encoder_GetMechAngle(void):** returns the mechanical angle.
- **Encoder_GetMechSpeed(void):** returns the mechanical Speed.

The main function is *Encoder_read(void)* which reads the counter TCNT and calculates the parameters. The algorithm is the one shown in Figure A.2. All the values can be changed in the file *encoder.c*, the value REV_DIR is a boolean parameter that allows us to change the direction that the encoder assumes as positive. ENC_EDG_REV is the number of edges per revolution which is equal to $4 \times n_{pul\_rev}$ and KS_ENC is a constant that converts the digital value into a speed in rad/sec.

During the experiments, we will use a motor with an incremental encoder because as described here, it is already implemented. However, the final implementation will be accomplished with an absolute encoder as the described in Figure 5.5

A.2 Firmware Overview

The YRotateItRX23T_V1 development board comes with libraries for motor control, transformation algorithms, A/D conversions, PWM configuration, etc. The software is written in C and it uses a propietary IDE called 'e2studio' which is a modified version of the open-source software Eclipse. includes a sample embedded firmware that implements a sensorless DC brushless control approach with PC communication for visualization and data logging. This software was modified for our purposes. In this section an overview of the principal parameters and files will be stated.

- **const_def.h**: here the frequencies for the current loop, velocity loop and PWM can be set. Several parameters like the dead time between switching of the power branches and the current and voltage conversion gain for the A/D are defined.

- **customize.h**: allows to enable or disable several characteristics like PI auto tuning, field weakening, motor identification, etc. For our tests, all of them were disabled so as to have a minimum software running. Also, the PI gain amplification factor, the parameters unit resolution and gain factors are defined. They are needed to convert from digital values into real numerical values.

- **def_par.h**: it contains the motor parameters like the number of poles, the maximum speed, the inductance and resistor values, the maximum current and the PI factors for the current and speed loop. It is important to note that the values established here are not the real ones, but digital equivalents that depends on the gain factors defined in **customize.h**. It is implemented in this way because when the software is connected to the PC, it can only receive values in integer numbers and not floats. Therefore, a gain factor from float to integer is needed.

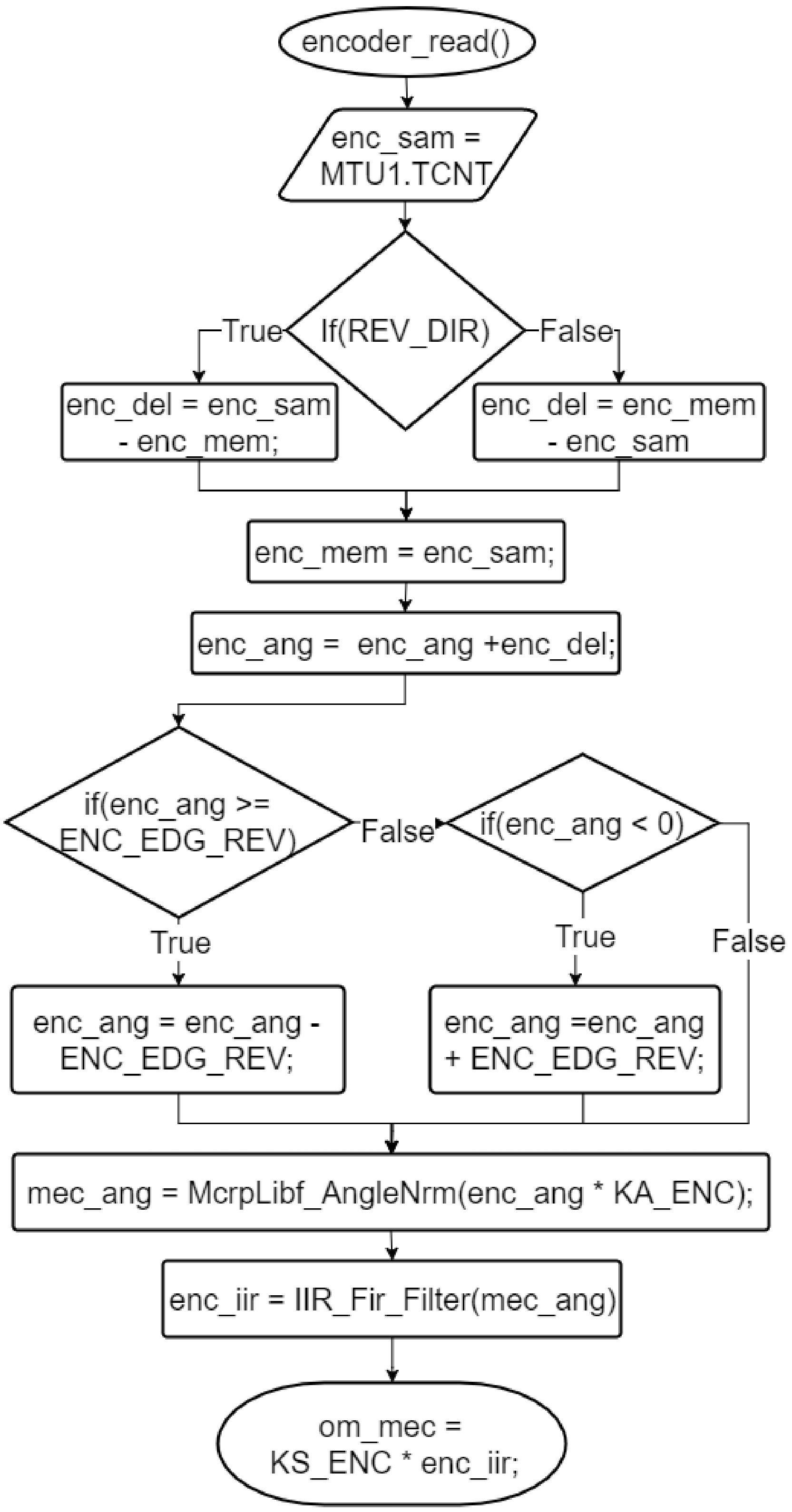

encoder_read()
enc_sam = MTU1.TCNT
If(REV_DIR)
True
False
enc_del = enc_sam - enc_mem;
enc_del = enc_mem - enc_sam
enc_mem = enc_sam;
enc_ang = enc_ang +enc_del;
if(enc_ang >= ENC_EDG_REV)
False
if(enc_ang < 0)
True
True
False
enc_ang = enc_ang - ENC_EDG_REV;
enc_ang =enc_ang + ENC_EDG_REV;
mec_ang = McrpLibf_AngleNrm(enc_ang * KA_ENC);
enc_iir = IIR_Fir_Filter(mec_ang)
om_mec = KS_ENC * enc_iir;

Figure A.2: Flow chart of the function encoder_read

- **globdef.h**: several global definitions are implemented here.
- **motorcontrol.h**: the libraries for the motor control are implemented here. It contains the main control loop, the PI controllers, the PWM and A/D converters inizialization, etc. It was the main part that we modified.
- **encoder.h**: here the encoder library is defined, contains the initialization parameters and the functions to get the angular position and speed.
- **pws_E6132.h**: this file has the characteristic of the power module stage, including non-linearities compensations and deadtime values. If the external power stage, also provided with the kit, is going to be used, it is necessary to replace this file.

A.2.1 PWM Generation and A/D Conversion

Since the RX23T is a microcontroller designed for motor control, it already contains functionalities for PWM waveform generation. In particular, the waveform is generated using the counters MTU3 and MTU4 in an operation mode called *Complementary PWM mode.* It is done in this way in order to implement the dead time value needed to avoid two branches to be set 'ON' at the same time.

The way in which it works is basically by having two counters MTU3.TCNT and MTU4.TCNT with an offset equal to the dead time defined. Then, the switching of a branch occurs only when both counters arrive to the reference value. In Figure A.3 we can see the behavior of a branch. MTU3.TCNT and MTU4.TCNT increase up to a value set by the register MTU3.TGRA. In practice, it defines the PWM carrier signal, the branch's duty cycle is given by the register MTU4.TGRA, when MTU3.TCNT is equal to MTU3.TGRA, the negative phase output changes its state. Due to the dead time offset, MTU4.TCNT reaches the same reference value a little later and switches the positive phase output. Then, the PWM signal can be generated simply by writing the desired duty cycle on the correspondent registers.

In the *YRotateItRX23T_V1*, the phases are called U,V and W. They are split in one high called H an another one low called L. Those phases are connected to the following pins in the RX23T:

Phase	Pin name	Duty Cycle Register
UH	MTIOC3B	MTU3.TGRD
UL	MTIOC3D	
VH	MTIOC4A	MTU4.TGRC
VL	MTIOC4C	
WH	MTIOC4B	MTU4.TGRD
WL	MTIOC4D	

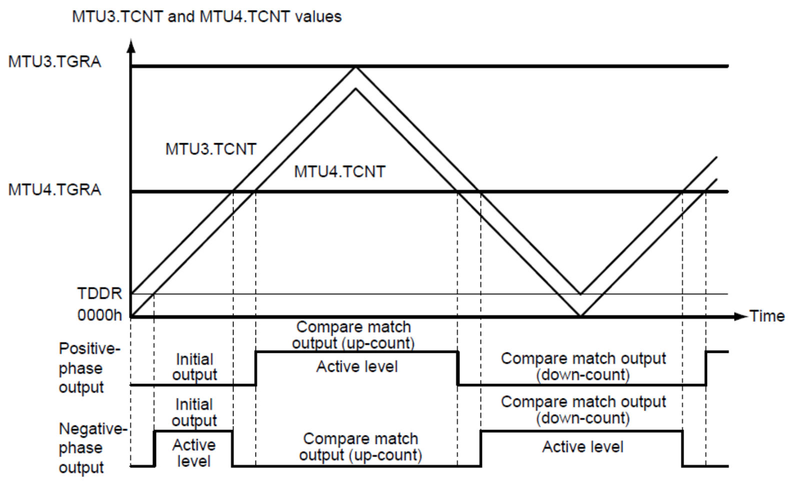

MTU3.TCNT and MTU4.TCNT values
MTU3.TGRA
MTU3.TCNT
MTU4.TCNT
MTU4.TGRA
TDDR
0000h
Time
Positive-phase output
Initial output
Compare match output (up-count)
Active level
Compare match output (down-count)
Negative-phase output
Initial output
Active level
Compare match output (up-count)
Compare match output (down-count)
Active level

Figure A.3: The PWM waveform generation implemented in the RX23T

In order to set a duty cycle, all we have to do is to modify the correspondent register.

The sampling frequency is equal to the A/D conversion frequency and it is related to the PWM frequency. In particular, it can be configured in the file **defpar.h** with the parameters *SAM_FRE_DEF* and *F_RATIO_DEF*, the first one defines the desired sampling frequency and the second the ratio, from which we can calculate the PWM frequency as:

$$pwm\_frequency = SAM\_FRE\_DEF \times F\_RATIO\_DEF \tag{A.2}$$

In practice, the A/D converter is configured to acquire a measure every time that MTU4.TCNT starts the down counting plus a delay that can be set by the user. However, it contains the 'Interrupt Skipping Function 1', in which up to 7 times an interruption request from the A/D converter can be skipped. Therefore, if for example, we have a selected sampling frequency of 8 kHz and a frequency ratio of 2, then the PWM frequency will be 16 kHz and the number of skip interrups will be equal to 1. This interruption request is important because it calls the main control function **MC_Conint()**. If the motor is aligned, then the reference for the quadrature current is given from the speed PI controller function *MC_SpeedPI*().

The library **motorcontrol.h** contains all the routines related to the configuration and control of the motor. The main functions are described:

- **MC_SetOff()**: it measures the initial offset currents.
- **MC_IniPar()**: initialization of parameters related to PWM, motor, A/D and main loop frequency
- **MC_IniPWM()** : initialization of the PWM and main interrupt. Configures the registers for the counters MTU3 and MTU4
- **MC_StartPWM()** : starts the PWM and the main interrupt loop. It does not start the PWM outputs.
- **MC_ConInt()** : it contains the main interrupt function. it is in charge of the current and speed control. it is called every interruption request of the A/D.
- **MC_adman():** read the phase currents and transforms them to its equivalents in alpha/beta frame using Clarke transformation.
- **MC_IdIqPI(idr,idm,iqr,iqm,vfmax,vdc,vqc,idint,iqint):** it implements the PI controllers for the Id and Iq currents. It takes the reference idr and iqr and the measurements idm and iqm, generating the control voltage outputs vdc and vqc. It also contains a saturation value given by vfmax and the address to save the integration terms idint and iqint.
- **MC_PWMGen():** it calculates the required duty cycles of the PWM from the reference voltage generated by the PI controllers.
- **SET_PWM(duty_u,duty_v,duty_w):** it sets the new PWM duty cycles in the output.
- **MC_SpeedPI(f_omrif,omegae,iqmax,errint):** it is the speed PI controller. It takes the reference speed f_omrif, the measured omegae and the maximum allowable current iqmax to generate the reference current iqr.

Finally, the board provides a proprietary library from Renesas called *mcrplibf.h* which contains several mathematical functions and transformations. The ones we will use are the following:

- **McrpLibf_alphabeta_dq(iam,ibm,idm,iqm):** converts the currents iam and ibm from alpha/beta frame to direct/quadrature frame. It is the Clarke transformation.
- **void McrpLibf_AngleSet(SystemPhase):** it sets the angular phase necessary to do the transformations and its inverses.
- **McrpLibf_dq_alphabeta(vdc,vqc,vac,vbc):** it is the equivalent to the Clarke anti transformations. Takes vdc and vqc and outputs vac and vbc.
- **McrpLibf_alphabeta_uv(vac,vbc,vuc,vvc):** the Park Transformation is applied in this function, taking vac, and vbc and converting them to the phase voltages vuc and vvc. It is important to note that it assumes that the three voltages are balanced, so the third phase current can be calculated from $vwc = vuc - vvc$.
- **McrpLib_uv_alphabeta(vuc,vvc,vac,vbc):** it does the inverse operation from the above function.

With all this information, we can now present the main control software algorithm which can be seen in Figure A.4, it is called each time interval with a frequency much faster than the electrical time constant.

What it does is basically the following: it reads the encoder to get the mechanical angle and speed, after it reads the phase currents and transform them into its direct/quadrature equivalents. It also reads the bus voltage to get the maximum admissible voltage. Then, it transforms the mechanical speed into the electrical speed by multiplying the first one times the number of pole pairs. If the motor is not aligned, i.e. if the offset angle is unknown, it sets the system phase to zero. If it is aligned then the system phase is simply the electrical angle plus the angle offset. After that, it sets the new electrical angle to do the calculations.

Then, we obtain the reference direct/quadrature voltages from the PI controllers and the algorithm transforms them back to phase voltages, afterwards it configures and updates the PWM values (if neccesary a speed filter is applied). Finally, if the system is not aligned, it performs the align() function, otherwise, it implements the speed PI controller.

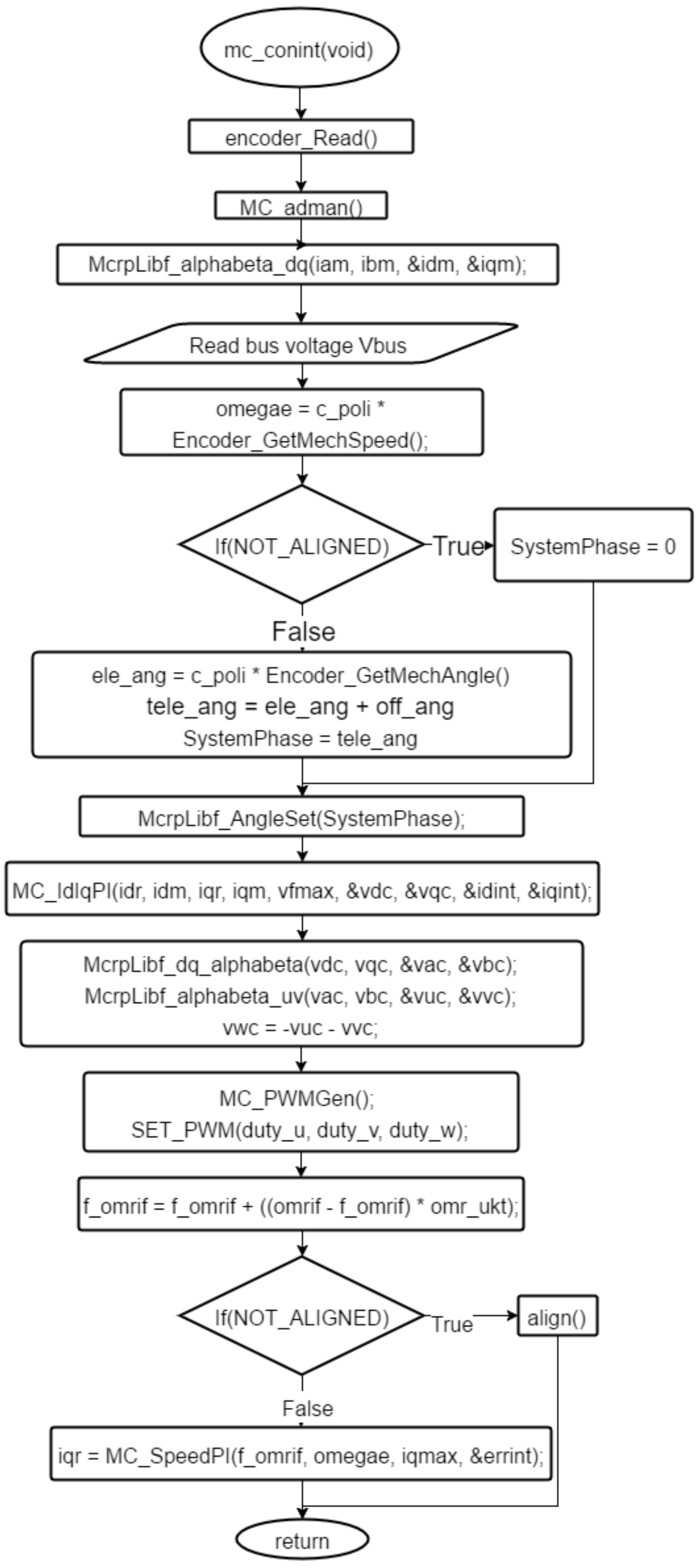

mc_conint(void)
encoder_Read()
MC_adman()
McrpLibf_alphabeta_dq(iam, ibm, &idm, &iqm);
Read bus voltage Vbus
omegae = c_poli * Encoder_GetMechSpeed();
If(NOT_ALIGNED)
True
SystemPhase = 0
False
ele_ang = c_poli * Encoder_GetMechAngle()
tele_ang = ele_ang + off_ang
SystemPhase = tele_ang
McrpLibf_AngleSet(SystemPhase);
MC_IdIqPI(idr, idm, iqr, iqm, vfmax, &vdc, &vqc, &idint, &iqint);
McrpLibf_dq_alphabeta(vdc, vqc, &vac, &vbc);
McrpLibf_alphabeta_uv(vac, vbc, &vuc, &vvc);
vwc = -vuc - vvc;
MC_PWMGen();
SET_PWM(duty_u, duty_v, duty_w);
f_omrif = f_omrif + ((omrif - f_omrif) * omr_ukt);
If(NOT_ALIGNED)
True
align()
False
iqr = MC_SpeedPI(f_omrif, omegae, iqmax, &errint);
return

Figure A.4: Main loop Flowchart.

Alignment Function

As we stated before, if the motor phase offset is unknown, it is necessary to perform an alignment function called align(). Which is implemented as a state machine that can be seen in Figure A.5.

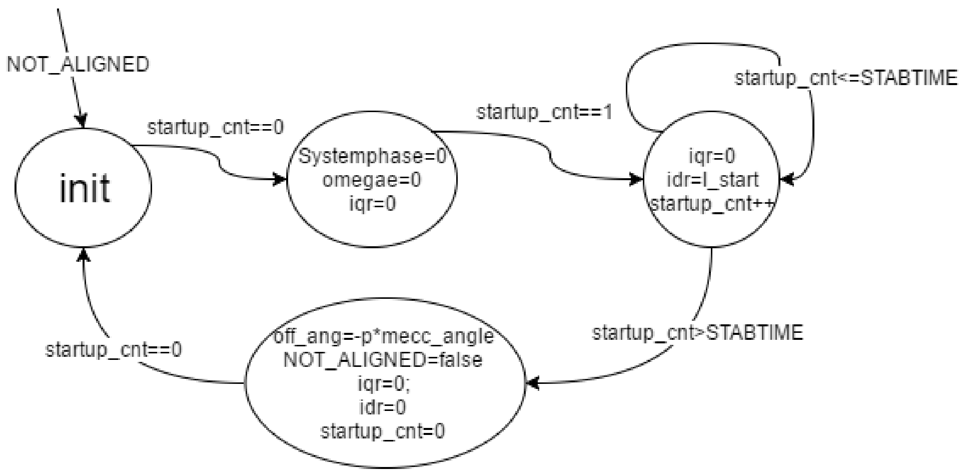


Figure A.5: Alignmnet procedure state machine.

If the motor is not aligned, it sets the phase system, the electrical speed and the reference Iq current. Then, it applies a reference Id current equal to a fixed value given by I_start which can be changed until the software reaches the 'Stabtime' variable, which is the amount of time that the motor needs to stay in the initial position, this is done in order to let pass any possible transients. Finally, it calculates the offset phase off_ang by multiplying the mechanical angle calculated times the number of poles p, it set NOT_ALIGNED to zero and resets all the currents. This procedure must be called each time that the system is initialized.

Appendix B

MSSI-040H-027D Datasheet

Definitionen / Definitions

Abk. Abbr.	Einheit Unit	Kenngrößen	*Characteristics*
T_{maxSt}	Ncm	**Stillstandsdrehmoment:** das für sehr kurze Zeit erreichbare, größte Moment bei Stillstand.	***Stall torque:*** *peak torque at standstill (very short time).*
T_{max}	Ncm	**Maximal nutzbares Moment:** begrenzt durch den maximalen Strom.	***Maximum usuable torque:*** *Limited by the maximum current.*
T_o	Ncm	**Dauer-Stillstands-Drehmoment:** Moment bei Stillstand, bei einer mittleren Temperaturerhöhung der Wicklung von 100K, bei einer max. Umgebungstemperatur von 40°C (nach VDE 0530 Isolierstoffklasse F).	***Continuous stall torque:*** *torque at standstill at a max. mean temperature of the windings of 100K, at a max. ambient temperature 40°C (104°F) (according to VDE 0530 insulation class F).*
T_n	Ncm	**Nenndrehmoment:** Motordrehmoment bei max. abgegebener Dauerleistung/Nennleistung.	***Nominal torque:*** *Motor torque at max. continuous power/nominal power.*
P_n	Watt	**Nennleistung:** mechanisch abgegebene Leistung bei Nennmoment und Nenndrehzahl.	***Nominal power:*** *mechanical motor power at nominal speed and nominal torque.*
n_n	min^{-1}, rpm	**Nenndrehzahl:** Motordrehzahl bei Nennleistung.	***Nominal speed:*** *motor speed at nominal power.*
n_o	min^{-1}, rpm	**Leerlaufdrehzahl:** max. erreichbare Motordrehzahl bei Nenn-Zwischenkreisspannung.	***No load speed:*** *max. achievable motor speed at nominal bus voltage.*
K_t	Ncm/A	**Drehmomentkonstante:** Verhältnis Motordrehmoment zu aufgenommenem Spitzenleiterstrom.	***Motor torque constant:*** *ratio of motor torque to current applied to the motor windings.*
I_{max}	A_{eff}	**Max. zulässiger Strom:** begrenzt durch die Wicklungserwärmung oder den Servo-Controller.	***Max. allowable motor current:*** *limited by the heating of the windings or by the servo controller.*
I_o	A_{eff}	**Dauer-Stillstands-Strom:** Motorwicklungsstrom, der das Stillstandsdrehmoment T_o erzeugt.	***Continuous stall current:*** *winding current that produces the continuous stall torque T_o.*
R_{tt}	Ω	**Anschlußwiderstand:** Widerstand, der bei 20°C Umgebungstemperatur zwischen zwei freien Anschlußleitungen gemessen wird.	***Connection resistance:*** *resistance measured at 20°C (68°F) ambient temperature between two phase for the motor winding.*
τ_E	ms	**Elektrische Zeitkonstante:** gibt das Verhalten der Motorwicklung im Stromregelkreis an. Sie entspricht dem Quotienten aus Anschlußinduktivität und Anschlußwiderstand: $\tau_E = L_{tt} / R_{tt}$.	***Electrical time constant:*** *describes the behaviour of the motor windings in the current control loop. It is the ratio of motor inductance to resistance: $t_E = L_{tt} / R_{tt}$.*
J	gcm^2	**Rotor-Massenträgheitsmoment:** polares Massenträgheitsmoment des Rotors.	***Rotor mass moment of inertia:*** *polar mass moment of inertia of the rotor.*
T_{max}/J	$1/s^2$	**Dynamikfaktor:** beschreibt die max. Beschleunigung des Rotors bei max. Moment, ohne Fremdlast.	***Dynamics factor:*** *describes the max. acceleration of the rotor at max. torque under no load condition.*
m	g	**Gewicht**	***Weight***

Umrechnungstabelle /
Conversion table:

1 Ncm = 0,0885 in.lb
$1\text{-}10^{-7}$ kgm^2 = 1 gcm^2 = 8.85 x 10^{-7} $in.lb.s^2$
1 kW = 1,34 hp

Schnellauswahl / Quick Search

Motor Baugröße / Motor Size

MSSI-017H-010D	0.7	1.5
MSSI-017H-020D	1.3	2.7
MSSI-017H-030D	1.7	3.8
MSSI-022H-007D	1.2	3.8
MSSI-022H-017D	2.5	9.7
MSSI-022H-027D	3.2	13.6
MSSI-032H-007D	2.6	9.5
MSSI-032H-017D	5.1	19.4
MSSI-032H-027D	7.6	24.5
MSSI-040H-007D	5.9	23.5
MSSI-040H-017D	11.4	46.2
MSSI-040H-027D	18	72.4
MSSI-055H-015D	25.5	70.2
MSSI-055H-030D	49.2	160
MSSI-055H-045D	71.9	230

▬ Dauermoment / *Continuous Torque* in Ncm
▭ max. Drehmoment / *max. Torque* in Ncm

Motor-Getriebe-Einheit / Motor-Gear-Unit

GCP 022 Standard

1-stufig / 1-stage	0.1	i = 4	6000
2-stufig / 2-stage	0.5	i = 16 / 28	6000
3-stufig / 3-stage	1.5	i = 64 / 112 / 196	6000

GCP 032 Standard

1-stufig / 1-stage	0.4	i = 4	5000
2-stufig / 2-stage	2	i = 12.08 / 18 / 25 / 32	5000
3-stufig / 3-stage	6	i = 64 / 130 / 200 / 256	5000

Spielarm / Low-Backlash

1-stufig / 1-stage	0.8	i = 4	5000
2-stufig / 2-stage	4	i = 12.08 / 18 / 25 / 32	5000
3-stufig / 3-stage	6	i = 64 / 130 / 200 / 256	5000

GCP 040 Standard

1-stufig / 1-stage	0.7	i = 4	5000
2-stufig / 2-stage	4	i = 12.25 / 20 / 25	5000
3-stufig / 3-stage	12	i = 64 / 120 / 184 / 293.89	5000

Spielarm / Low-Backlash

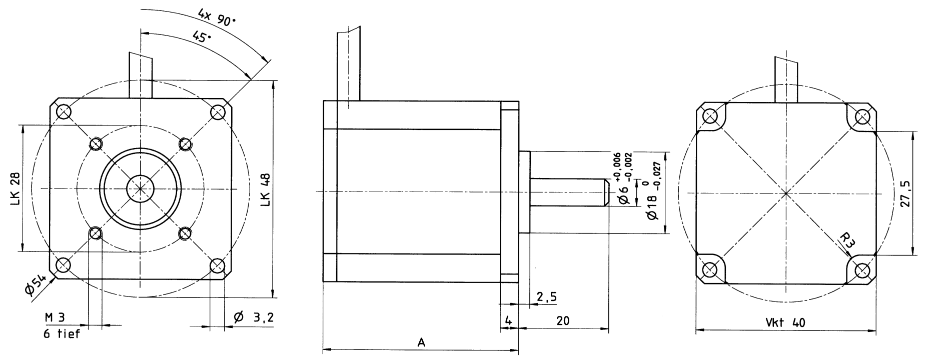
1-stufig / 1-stage	1.4	i = 4	5000
2-stufig / 2-stage	8	i = 12.25 / 20 / 25	5000
3-stufig / 3-stage	12	i = 64 / 120 / 184 / 293.89	5000

GCP 050 Spielarm / Low-Backlash

1-stufig / 1-stage	5.2	i = 5 / 25 / 50	4000
2-stufig / 2-stage	5.2	i = 10 / 100	4000

▬ Nenndrehmoment / *Nominal Torque* in Nm
▭ Nenndrehzahl / *Nominal Input Speed* in Nm

MSSI 040H mit Hallkommutierung / *MSSI 040H with hall commutation*

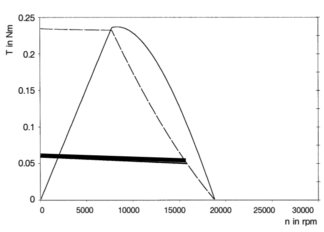


Baugröße / *Size*	**A**
MSSI-040H-007D	47
MSSI-040H-017D	57
MSSI-040H-027D	67

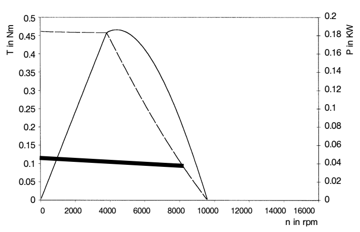
Radialkraft / *Radial Force*
Axialkraft / *Axial Force*
Siehe Kapitel „Zulässige Lagerbelastungen"/
See chapter „Premissible loads on bearing"

Motorkennlinien / Technische Daten *Motor characteristics / Technical Data*

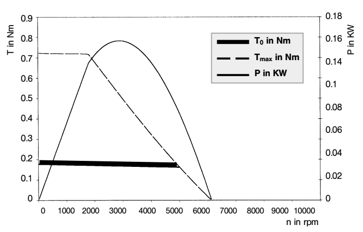
MSSI-040x-007x-024x



MSSI-040x-017x-024x



MSSI-040x-027x-024x



Kenngrößen / *Characteristics*	**Abk. *Abbr.***	**Einheit *Unit***	**MSSI-040H-**		
Baugröße / *Size*			-007D	-017D	-027D
Zwischenkreisspannung / *DC Bus Voltage*	U_D	V	24	24	24
Max. Drehmoment / *Max. Torque*	T_{max}	Ncm	23.5	46.2	72.4
Max. Strom / *Max. Current*	I_{max}	A_{eff}	16.0	16.0	16.0
Dauerstillstandsdrehmoment/*Continuous Stall Torque*	T_0	Ncm	5.9	11.4	18.0
Dauerstillstandsstrom / *Continuous Current*	I_0	A_{eff}	4.0	3.9	3.9
Leerlaufdrehzahl / *No-Load Speed*	n_0	min^{-1}	19.100	9.700	6.200
Nenndrehmoment / *Nominal Torque*	T_n	Ncm	5.0	9.7	16.3
Nennstrom / *Nominal Current*	I_n	A_{eff}	3.6	3.5	3.7
Nenndrehzahl / *Nominal Speed*	n_n	min^{-1}	15.900	8.200	5.100
Drehmomentkonstante / *Torque Constant*	K_t	Ncm/A	1.3	2.5	3.8
Wicklungstemperatur / *Winding temperature*	θ_{max}	°C	140	140	140
Umgebungstemperatur / *Ambient temperature*	θ_u	°C	40	40	40
Wärmeübergangswiderstand/*Thermal Resistance*	R_{th}	K/W	4.5	3.9	3.25
Anschlußwiderstand / *Terminal Resistance*	R_{tt}	Ω	0.5	0.48	0.6
Anschlußinduktivität / *Terminal Inductance*	L_{tt}	mH	0.2	0.3	0.4
Elektrische Zeitkonstante/*Electrical Time Constant*	τ_E	ms	0.4	0.6	0.7
Dynamikfaktor / *Dynamics Factor*	T_{max} J	$1/s^2$	280.000	300.000	310.000
Massenträgheitsmoment / *Mass Moment of Inertia*	J	gcm^2	8.5	15.5	23.5
Masse / *Weight*	m	g	185	265	340

Zulässige Lagerbelastungen/Permissible load on bearing

**Zu den Kennlinien der max. Radialkraft
müssen folgende Punkte berücksichtigt werden.**

MSSI 017H
Lagerlebensdauer: 20.000 h.
Kraftangriffspunkt der Radialkraft bezogen auf die Wellenmitte.
Axialkraft 3N bezogen auf die Wellenachse.

MSSI 022H / MSSI 032H
Lagerlebensdauer: 20.000 h.
Kraftangriffspunkt der Radialkraft bezogen auf die Wellenmitte.
Axialkraft 5N bezogen auf die Wellenachse.

MSSI 040H / MSSI 055H
Lagerlebensdauer: 20.000 h.
Kraftangriffspunkt der Radialkraft bezogen auf die Wellenmitte.
Axialkraft 10N bezogen auf die Wellenachse.

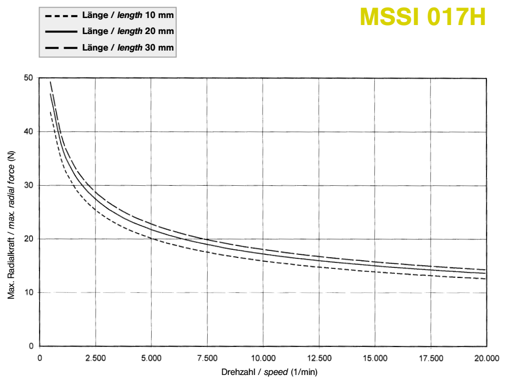
Please take the following into account when analyzing the characteristic curves of the max. radial force

MSSI 017H
Service life of bearing: 20.000 h.
Contact point of force related to the center of the shaft.
Axial force 3N related to the shaft axis.

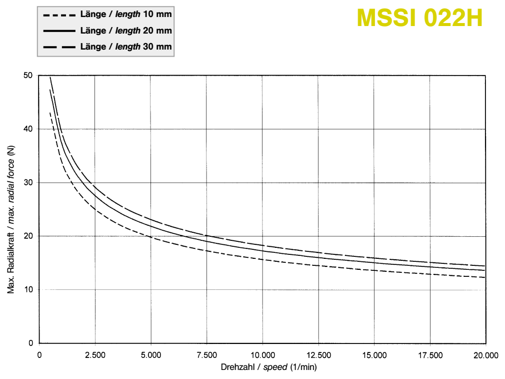
MSSI 022H / MSSI 032H
Service life of bearing: 20.000 h.
Contact point of force related to the center of the shaft.
Axial force 5N related to the shaft axis.

MSSI 040H / MSSI 055H
Service life of bearing: 20.000 h.
Contact point of force related to the center of the shaft.
Axial force 10N related to the shaft axis.

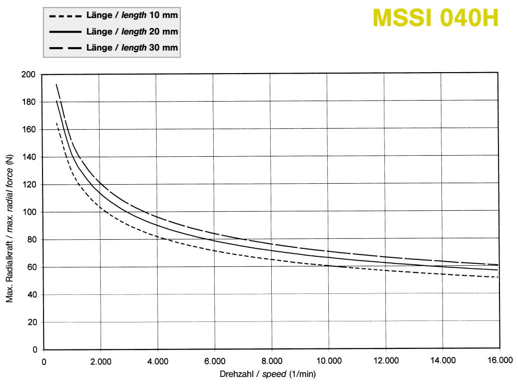
MSSI 017H



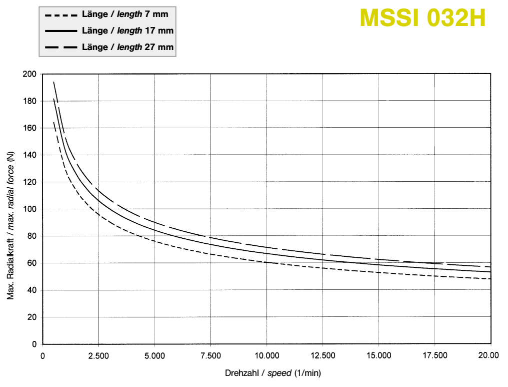
MSSI 022H



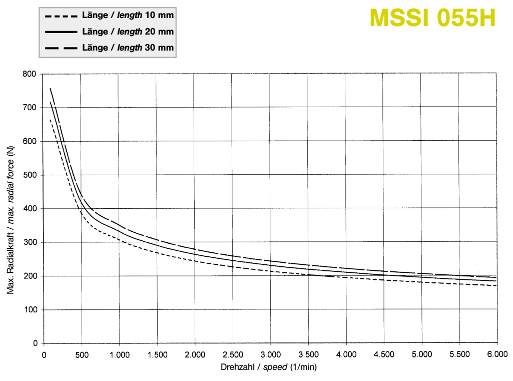
MSSI 040H



MSSI 032H



MSSI 055H



Appendix C

Orbis true absolute rotary encoder datasheet

Preliminary product information
OrbisP01_03
Issue 3, 7th April 2016

Orbis™ true absolute rotary encoder

Orbis™ is a true absolute rotary encoder suitable for applications where a typical OnAxis encoder cannot be mounted at the end of the rotating shaft due to space constraints.

The encoder comprises a diametrically magnetized permanent ring magnet and a printed circuit board. Geometric arrangement of 8 RLS' proprietary Hall sensors on a PCB enables generation of one period of sine and cosine signals per mechanical magnet revolution. Moreover, it also enables cancellation of third harmonic component that becomes non-negligible at low magnet ride height.

An adaptive filtering function ensures high resolution at low rotation speeds and low angle phase delay at high rotational speeds. Orbis™ also features an additional built-in self-calibration algorithm that improves encoder's accuracy after installation.

Orbis™ through-hole measuring principle allows customisation with various board and magnet sizes to suit your application.

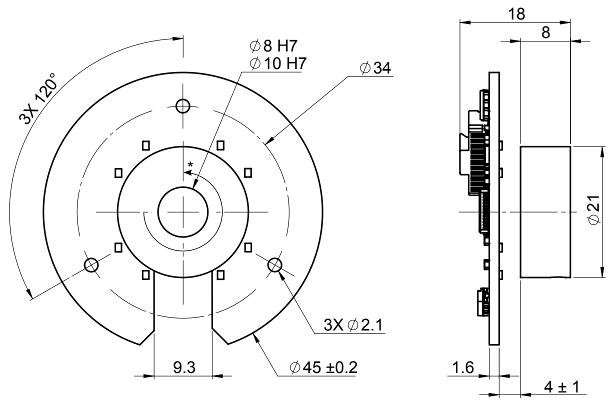
- **True absolute encoder**
- **14 bit resolution**
- **8 proprietary Hall sensor ASICs**
- **Through-hole design enables its mounting anywhere along the shaft**
- **Self-calibration after assembly**
- **No magnetic hysteresis**
- **Buit-in self-diagnostics**
- **Status LED**
- **SPI, SSI, BiSS-C, PWM, and asynchronous serial communication**
- **Wide installation tolerances**

A RENISHAW associate company

Preliminary product information
OrbisP01_03

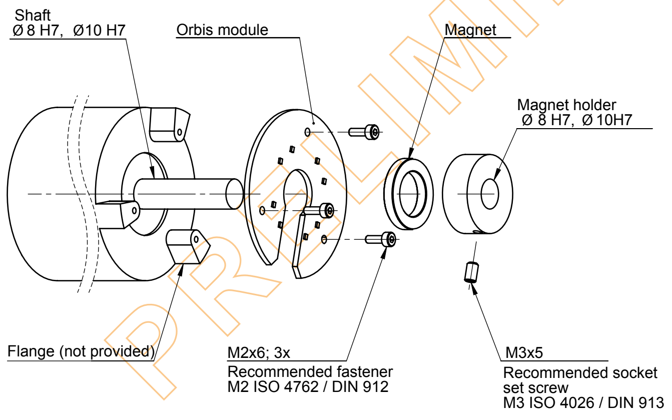
Dimensions

Dimensions and tolerances in mm.



*CCW - positive measuring direction

Installation drawing



Technical specifications

System data		
Reading type	Axial reading	
Resolution	14 bit	
Maximum speed	10,000 rpm	
Encoder accuracy	< ±0.3° (before installation - errors caused by mounting inaccuracy are not included)	
Hysteresis	No magnetic hysteresis	
Repeatability	< ±2 LSB	
Electrical data		
Supply voltage	4 V to 6 V	
Set-up time	< 2 ms	
Power consumption	< 80 mA	
Output load	PWM, SPI	Max. ±20 mA at 3.3 V
	RS422	120 mA short term, 60 mA limited
ESD protection	HBM, Class 2, max. ±2 kV (as per Mil-Std 883 Method 3015.7)	
Mechanical data		
Available magnet size	OD: 19 mm, ID: 12 mm, thickness: 3 mm (other magnet sizes possible)	
Available adapter sizes (inner diameter)	8 mm	
	10 mm	
Encoder outer diameter	45 mm (customisation possible)	
Encoder hole diameter	13 mm (customisation possible)	
Environmental data		
Operating temperature	–40 °C to +105 °C	

Status indicator LED

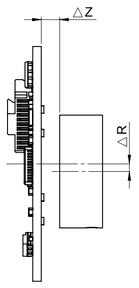
The LED provides visual feedback of signal strength, error condition and is used for set-up and diagnostic use.

LED	Status
Green	Normal operation; position data is valid.
Orange	Warning; position is valid, but some operating conditions are close to limits.
Red	Error; position data is not valid.
No light	No power supply.

Installation instructions

Installation tolerances
Precise magnet and board positioning is key to achieving good overall accuracy.

Axial (ΔZ) displacement (ride height)	4 mm nominal ±1 mm
Radial (ΔR) displacement	0.3 mm

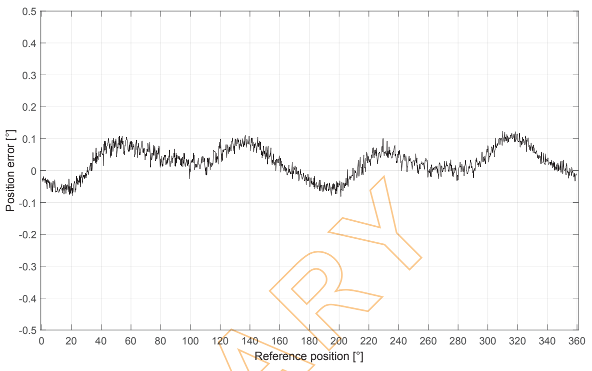


Axial position adjustment (ride height)

The nominal gap between the permanent magnet and the printed circuit board (opposite side of connector) is 4 mm ±1 mm. Any non-magnetic tool with 4 mm thickness can be used to check the correct ride height setting mechanically.
The integrated LED can be used as a coarse indicator. When the correct ride height is achieved, the LED glows green and does not change colour when the magnet rotates.

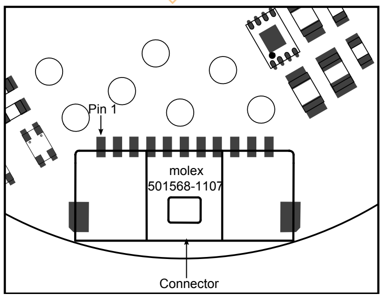
A typical accuracy plot after good installation of Orbis encoder is shown in the graph on the right.

For highest accuracy options contact RLS.



Electrical connections

Pin	Wire Colour	Asynchronous serial	PWM	SSI	BiSS-C	SPI
1	Brown	5 V supply	5 V supply	5 V supply	5 V supply	5 V supply
2						
3	White	0 V (GND)	0 V (GND)	0 V (GND)	0 V (GND)	0 V (GND)
4						
5	Pink	-	-	-	-	-
6	Grey	-	-	-	-	-
7	Red	RX data in+	Status	Clock+	MA+	SCK
8	Blue	RX data in–	-	Clock–	MA–	$\overline{\text{CS}}$
9	Cable Shield	Cable Shield	Cable Shield	Cable Shield	Cable Shield	Cable Shield
10	Green	TX data out+	PWM Out	Data+	SLO+	MISO
11	Yellow	TX data out–	-	Data–	SLO–	MOSI

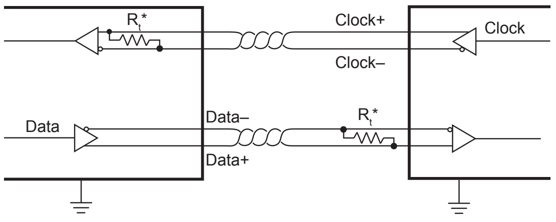


Communication interfaces

SSI - Synchronous serial interface

The encoder position, in 14 bit natural binary code, and the encoder status are available through the SSI protocol. The position data is left aligned. After the position data there are two general status bits followed by the detailed status information.

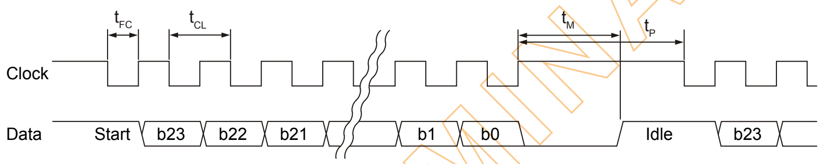
Electrical connection



Line signals	
Clock+	Clock non-inverted signal
Clock–	Clock inverted signal
Data+	Data non-inverted signal
Data–	Data inverted signal

* The Clock and Data lines are 5 V RS422 compatible differential pairs. The termination resistor on the Clock line is integrated inside the encoder. On the controller's side of Data line it should be added by the user or enabled in the controller.

SSI timing diagram



The controller requests the position and status data of the encoder by sending a pulse train to the Clock input. The Clock signal always starts from high. The first falling edge of the Clock latches the last position data available and on the first rising edge of the Clock the most significant bit (MSB) of the position is transmitted to the Data output. The Data output should then be read on the following falling or rising edge. On subsequent rising edges of the Clock signal the next bits are transmitted.

After the transmission of the last bit the Data output goes to low. When the t_M time expires, the Data output goes high. The Clock signal must remain high for at least t_P before the next reading can take place.

While reading the data, the half of a Clock period t_{CL} must always be less than t_M. However, reading the encoder position can be terminated at any time by setting the Clock signal to high for the duration of t_M.

Communication parameters

Parameter	Symbol	Min	Typ	Max
Clock period	t_{CL}	2 µs (400 ns *)		15 µs
Clock frequency	f_{CL}	70 kHz		500 kHz (2.5 MHz *)
Delay first clock	t_{FC}	1.25 µs		
Transfer timeout	t_M		16 µs	
Pause time	t_P		20 µs	

* With *Delay First Clock* function of the controller.

Appendix D

YROTATE-IT-RX23T User Manual

YROTATE-IT-RX23T

Rotate it! – Motor Control RX23T

UM-YROTATE-IT-RX23T
Rev. 1.0
October 6, 2015

Introduction

The Renesas Motor Control Kit called YROTATE-IT-RX23T, is based on the RX23T device from the powerful 32-bit RX microcontrollers family running at 40MHz and delivering up to 80DMIPs.

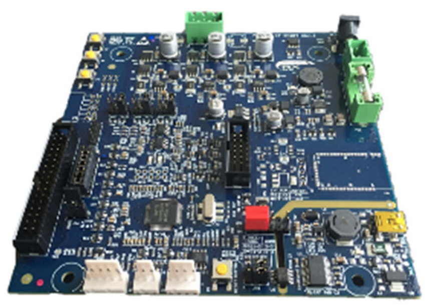
The kit enables engineers to easily test and evaluate the performance of the RX23T in a laboratory environment when driving any 3-phase Permanent Magnet Synchronous Motor (e.g. AC Brushless Motor) using an advanced sensorless Field Oriented Control algorithm. Typical applications for this type of solution are compressors, air conditioning, fans, air extractors, pumps, home appliances inverters and industrial drives.

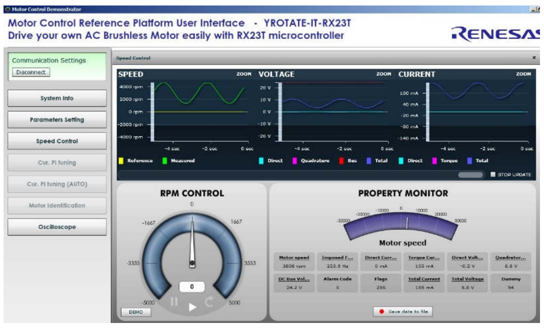
The phase current measurement is done via three shunts which offers a low cost solution, avoiding the need for an expensive current sensor or hall sensor. A single shunt current reading method is also available to ensure an even more compacter bill of material.

The powerful user-friendly PC Graphical User Interface (GUI) gives real time access to key motor performance parameters and provides a unique motor auto-tuning facility. Furthermore, it becomes also possible to select the best switching frequency and control frequency (e.g. control loop) to adapt the control dynamics suitable to the application requirements.

The hardware is designed for easy access to key system test points and for the ability to hook up to an RX23T debugger known as E1. Although the board is normally powered directly from the USB port of a Host PC, connectors are provided to utilise external power supplies where required.

The YROTATE-IT-RX23T is an ideal tool to check out all the key performance parameters of your selected motor, before embarking on a final end application system design.





Motor Control Reference Platform User Interface - YROTATE-IT-RX23T
Drive your own AC Brushless Motor easily with RX23T microcontroller

Motor speed	Imposed F...	Direct Curr...	Torque Cur...	Direct Volt...	Quadratur...
3608 rpm	253.9 Hz	0 mA	155 mA	-0.2 V	8.8 V
DC Bus Vol...	Alarm Code	Flags	Total Current	Total Voltage	Dummy
24.2 V	0	256	155 mA	8.8 V	54

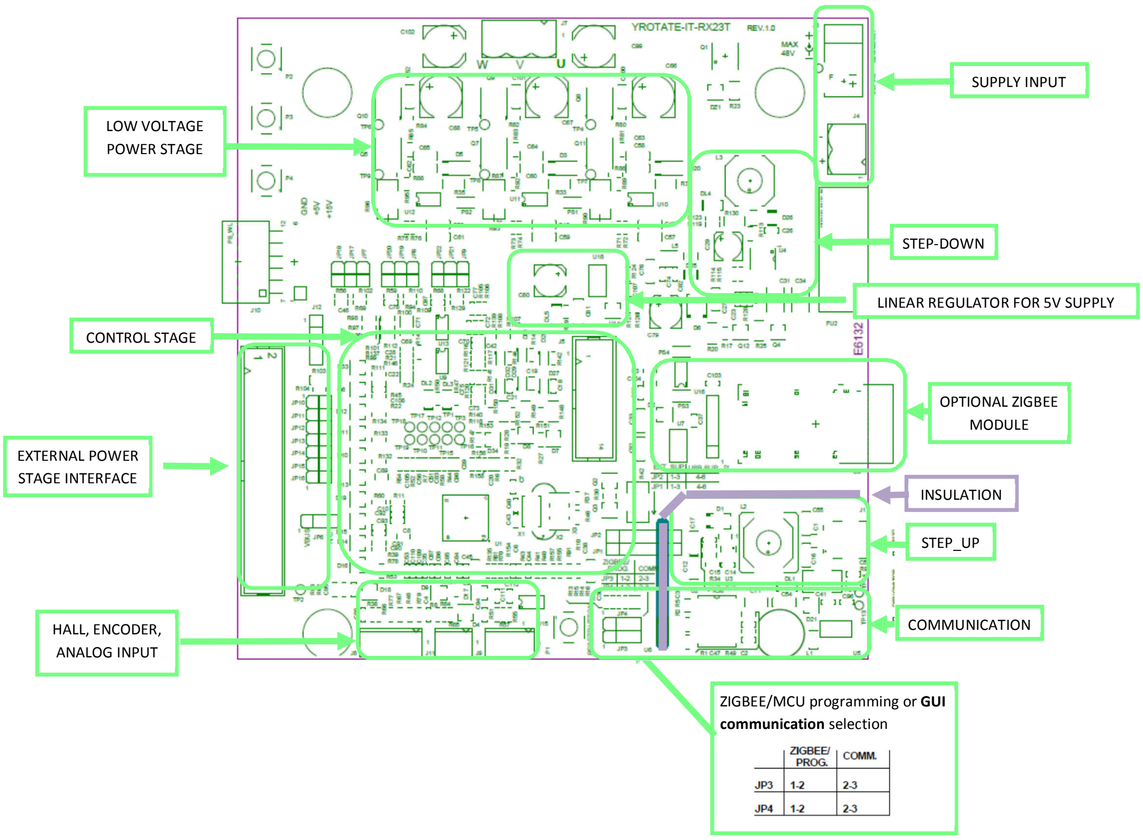
Target Device: **RX23T Microcontrollers Family**

1. Specifications & Hardware overview

ITEM	SPECIFICATIONS
TYPE OF MOTORS SUPPORTED	3-phase Permanent Magnet Synchronous (PMSM, PMAC, BLAC) 3-phase Brushless DC (BLDC)
KIT MOTOR PARTNAME	NANOTEC DB42S03, 24V_{DC}, 4000 RPM
KIT MAX INPUT RANGE	External power supply from: 20V_{DC} to 48$V_{DC,}$ 6A_{peak}
TRANSISTOR USED	Renesas Mosfets: RJK0654DPB, 60V, 30A
POWER SUPPLY OPTION	Either USB connection or external supply: up to 48V_{DC}
CURRENT DETECTION	One or three shunts configuration (10mΩ)
USB IC USED ON THE BOARD	FT232R - USB UART IC from FDTI, 76.6KBd communication speed
MICROCONTROLLER	RX23T (R5F523T5ADFM), 64-pin LFQFP, 40MHz, 64KB Flash, 10KB RAM
MCU PERFORMANCE	40MHz, 80DMIPs, 166 CoreMark
KEY FEATURES	Floating Point Unit, 3-phase inverter Timer 12-bit A/D Converter, fast on-chip Comparators, Port Output Enable
MCU EMBEDDED FIRMWARE	Sensorless vector control algorithm (Field Oriented Control)
SWITCHING FREQUENCY	4KHz to 64KHz, 16KHz by default (PWM frequency)
CONTROL LOOP FREQUENCY (SAMPLING FREQUENCY)	4KHz to 16KHz, 8KHz by default
CONTROL LOOP TIMING	50µs including debug features and auto-tuning/self-identification 40µs including only the sensorless vector control algorithm
CODE SIZE IN FLASH / RAM	24KB / 3KB
TOOL USED, VERSION	e^2studio 4.0.2.008. , CS+ , RXC Toolchain, CC RX version v2.03.00
COMPILER OPTIMIZATION LEVEL	Maximum, optimize for speed, little-endian data, RX V2 Core enable
ENVIRONMENT STANDARDS	RoHS compliant including China regulations WEEE, RoHS

To achieve these aims, three different DC-DC converters are used:

1. A step-up DC-DC converter to increase the voltage from the USB standard (5V) up to 13.5V_{DC}
2. A step-down converter and a low dropout linear converter, from the DC bus first to 15V and then to the CPU supply voltage: 5V, please find below the PCB overview.



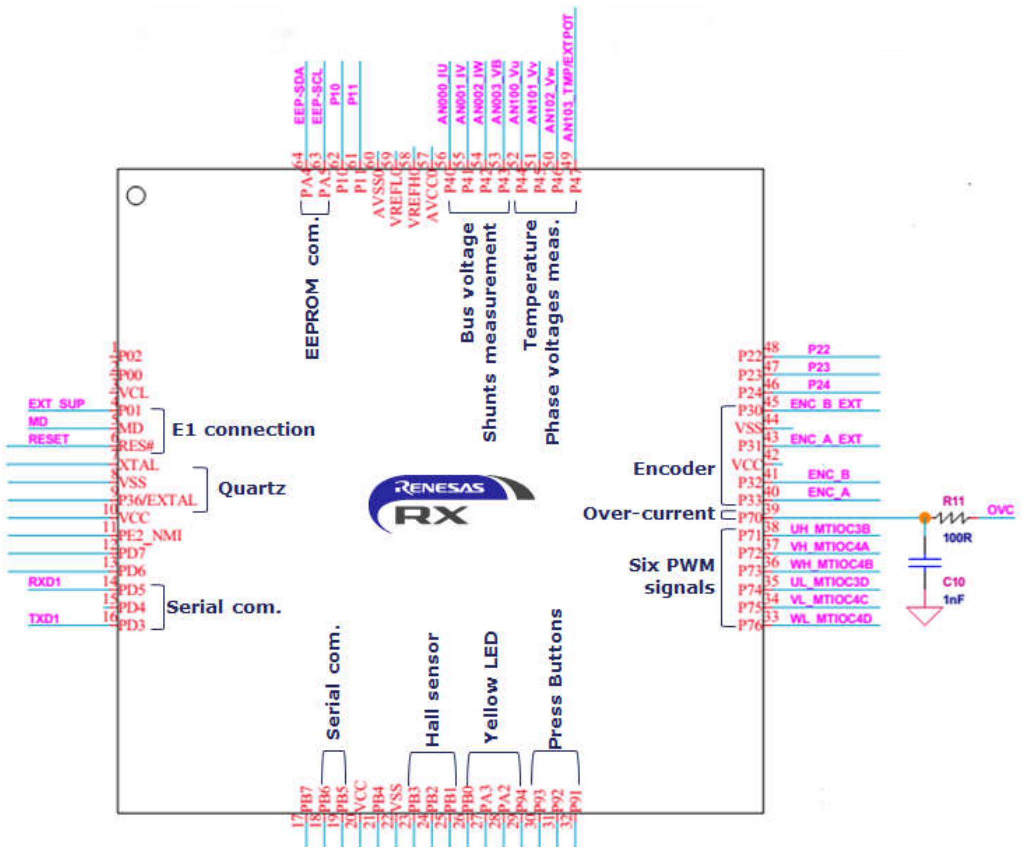
ZIGBEE/MCU programming or **GUI communication** selection

	ZIGBEE/ PROG.	COMM.
JP3	1-2	2-3
JP4	1-2	2-3

The full schematics and CAD design files (e.g. Gerber) of the inverter kit are available on the website: www.renesas.eu/motorcontrol , in the section related to the YROTATE-IT-RX23T development kit.

In the YROTATE-IT-RX23T kit, the RX23T in a 64-pin package was selected to ensure the management of inverter, external communications, three shunts, the EEPROM communication, the E1 debugger, three voltages phases, the over-current detection, the Bus voltage monitoring, etc.

The picture below is showing the detailed I/O pins assignment of the RX23T to manage the complete kit.

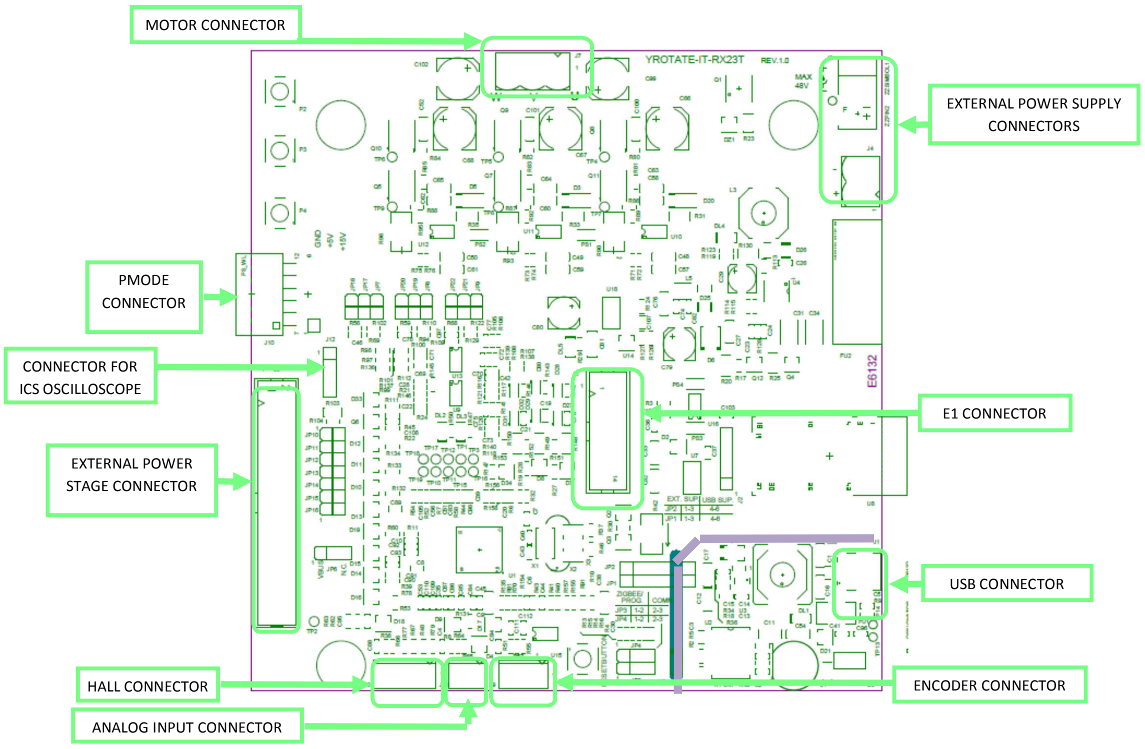

EEP-SDA
EEP-SCL
P10
P11
AN000_IU
AN001_IV
AN002_IW
AN003_VB
AN100_Vu
AN101_Vv
AN102_Vw
AN103_TMP/EXTPOT
EEPROM com.
Bus voltage
Shunts measurement
Temperature
Phase voltages meas.
EXT_SUP
MD
RESET
RXD1
TXD1
E1 connection
Quartz
Serial com.
P22
P23
P24
ENC_B_EXT
ENC_A_EXT
ENC_B
ENC_A
Encoder
Over-current
Six PWM signals
UH_MTIOC3B
VH_MTIOC4A
WH_MTIOC4B
UL_MTIOC3D
VL_MTIOC4C
WL_MTIOC4D
R11
100R
OVC
C10
1nF
Serial com.
Hall sensor
Yellow LED
Press Buttons

2. Connectors description

As in the following figure, you can find the position and the description of the connectors present on the board. Please refer to the board schematics for the full description of the connectors.

The E1 connector is used for the programming and the debugging of the software running on the RX23T. It can be connected either to the E2studio and the CS+ development environments.

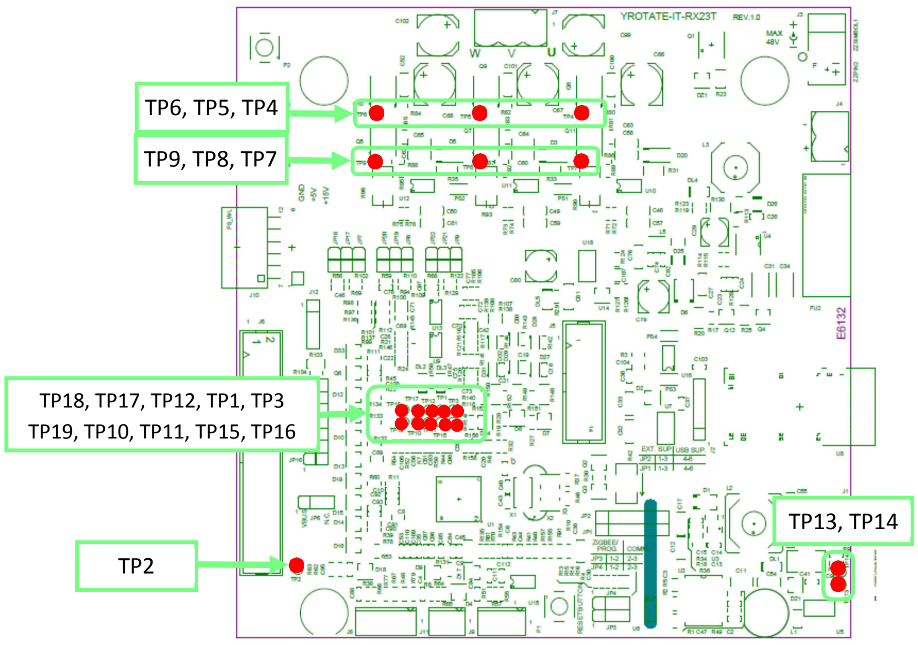
The external power stage connector is compatible with the power stages, designed for Renesas inverter kits, which are able, the first one to drive 230V$_{AC}$ motor up to 1.5KW, and the second one up to 60V$_{DC}$, 60A$_{DC}$. The schematics and Gerber file of the power stage are available on the website: www.renesas.eu/motorcontrol



5. Test points for debugging

Several specific test points are available on the board to visualize with the oscilloscope the behaviour of some internal analog signals.

Furthermore, it is possible to visualize internal variables as analog waveforms using filtered PWM outputs. Finally, it is very useful during the tuning process for adapting the software to a new motor to use the test points.



Please find below the description of the test points:

- **TP13**, **TP14**: are connected to the two USB communication signals, for debug purposes. Please refer to the board schematics for more details.
- **TP4**, **TP5**, and **TP6**: they are connected to the three inverter outputs (sources of the higher switches)
- **TP7**, **TP8**, and **TP9**: they are connected to the sources of the lower switches of the inverter
- **TP1, TP3, TP12, TP15, TP16**: they are connected to some microcontroller ports
- **TP17**, **TP18**, **TP19 are connected to the board GND**
- **TP10**, **TP11** are two filtered PWM outputs which can be used to visualize the behaviour of internal variables
- **TP2 not used**

12. Microcontroller RX23T short overview

The RX23T Group is 32-bit microcontroller and suited for single inverter control and has a built-in FPU (floating-point processing unit) that enables it to easily program complex inverter control algorithms. This helps to greatly reduce the man-hours required for software development and maintenance. Furthermore, thanks to the RX200 core the current consumed in software standby mode with RAM retention is a mere 0.45µA. RX23T MCUs operate in a broad voltage range from 2.7 V to 5.5 V, which is useful for inverter control, and are highly compatible with the RX62T Group at the pin arrangement and software level.

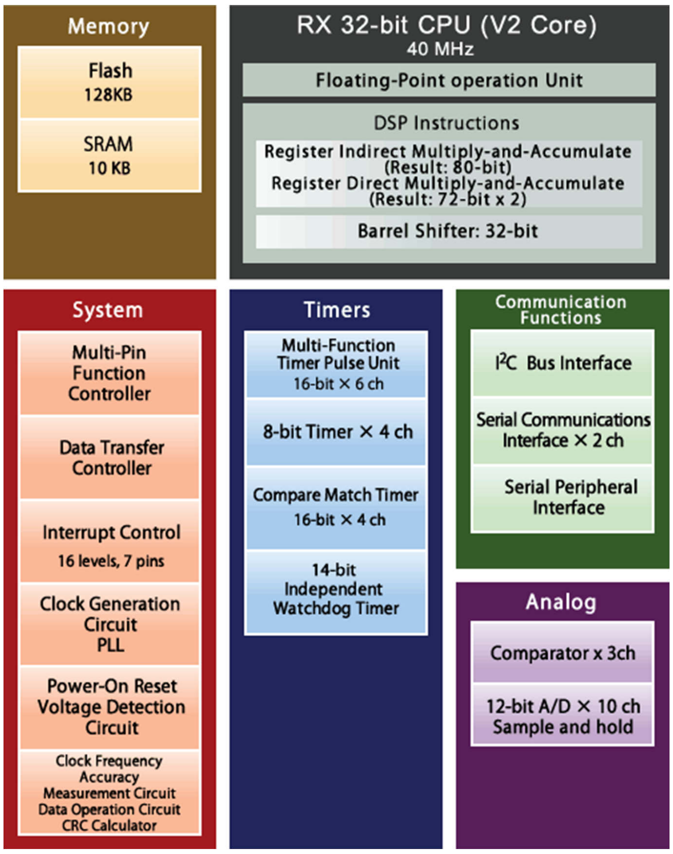
The main specifications of the RX23T microcontrollers are as follows:

Item	**RX23T Group**
CPU core	RX CPU v2 core running at 40MHz, delivering 2DMIPs/MHz General registers: 32-bit x 16, 32-bit multiplier & Divider Two types of Multiply-accumulator memory-to-memory operations and register-to-register operations
Power supply voltage	2.7 to 5.5V
Floating-point operation unit	Single-precision floating-point operation unit
Flash / RAM memory	Max.128 KB / 10KB

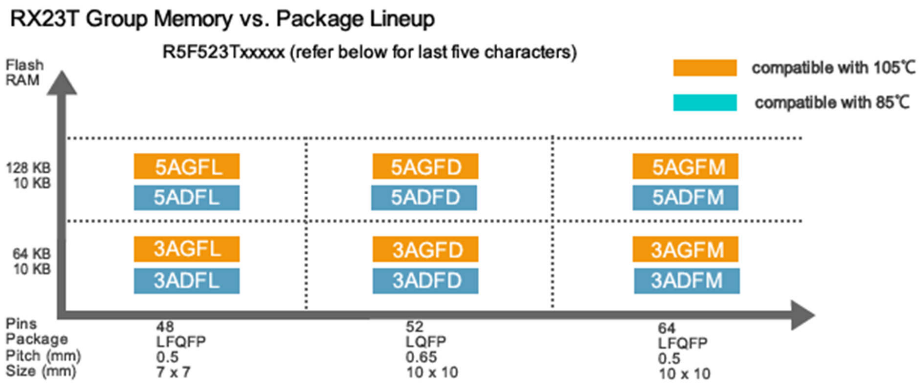
On-chip peripheral functions	**Transfer**	Data transfer controller (DTCa)
	Timers	Multi-function timer pulse unit 3: 16-bit x 6 channels (MTU3c) Port output enable 3 (POE3b) 8-bit timer (TMR): 8-bit x 2 channels x 2 units Compare match timer (CMT): 16-bit x 2 channels x 2 units
	Communications	Serial communications interface (SCIg): 2 channels I2C bus interface (RIICa): 1 channel Serial peripheral interface (RSPIa): 1 channel
	Analog	12-bit A/D converter (S12ADE): 10 channels Comparator C (CMPC): 3 channels D/A converter for generating ref. voltage for comparator C (DA)
	Safety	Clock frequency accuracy measurement circuit (CAC) Data operation circuit (DOC) 14-bit independent watchdog timer (IWDTa) A/D converter self-diagnostic/open-circuit detection function CRC calculator (CRC) Register write protection
	Clock generation circuit	Main clock oscillator Low-speed on-chip oscillator (LOCO) PLL frequency synthesizer Dedicated on-chip oscillator for the IWDT
	Other	Multi-function pin controller (MPC) Power-on reset circuit (POR) Voltage detection circuit (LVDAb)

Packages		PLQP0048KB-A(48-pin,LFQFP,7 x 7mm，0.5mm pitch)
		PLQP0052JA-A(52-pin,LQFP,10 x 10mm，0.65mm pitch)
		PLQP0064KB-A(64-pin,LFQFP,10 x 10mm，0.5mm pitch)

Please find below the RX23T microcontroller block diagram.


Memory
Flash 128KB
SRAM 10 KB
RX 32-bit CPU (V2 Core)
40 MHz
Floating-Point operation Unit
DSP Instructions
Register Indirect Multiply-and-Accumulate (Result: 80-bit)
Register Direct Multiply-and-Accumulate (Result: 72-bit x 2)
Barrel Shifter: 32-bit
System
Multi-Pin Function Controller
Data Transfer Controller
Interrupt Control 16 levels, 7 pins
Clock Generation Circuit PLL
Power-On Reset Voltage Detection Circuit
Clock Frequency Accuracy Measurement Circuit
Data Operation Circuit
CRC Calculator
Timers
Multi-Function Timer Pulse Unit 16-bit × 6 ch
8-bit Timer × 4 ch
Compare Match Timer 16-bit × 4 ch
14-bit Independent Watchdog Timer
Communication Functions
I²C Bus Interface
Serial Communications Interface × 2 ch
Serial Peripheral Interface
Analog
Comparator x 3ch
12-bit A/D × 10 ch Sample and hold

Please find below the memory line-up including the part-names.


RX23T Group Memory vs. Package Lineup
R5F523Txxxxx (refer below for last five characters)
compatible with 105℃
compatible with 85℃
Flash RAM
128 KB 10 KB
64 KB 10 KB
5AGFL
5ADFL
5AGFD
5ADFD
5AGFM
5ADFM
3AGFL
3ADFL
3AGFD
3ADFD
3AGFM
3ADFM
Pins Package Pitch (mm) Size (mm)
48 LFQFP 0.5 7 x 7
52 LQFP 0.65 10 x 10
64 LFQFP 0.5 10 x 10