End of Degree Project

Bachelor's degree in Industrial Technology Engineering

Hardware design and implementation of two-wheeled vehicle robots for a platooning system

Author: Josep Oriol Torta Valmaña

Director: Arnau Dòria-Cerezo
Víctor Repecho del Corral

Call: June 2017

ETSEIB

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

UPC

Summary

This End of Degree Project is centered on the building of a two-wheeled vehicle robot and the design and programming of current control for its motors. The project continues the task of three previous works regarding the same type of robot, trying to improve its hardware optimization and changing the controls.

The first part of the project is a description of the hardware implementation. All the components are characterized and their circuit schematics are documented and explained. Current sensors for the motors are designed, calculated and implemented in order to obtain the signal needed for executing the current control.

The second part explains the model of the vehicle and motors and the control structures that manage the entire robot. It is focused on the design of the motors control using the current as the main controlling variable. Some tests are showed to demonstrate the effectiveness of the controls.

Besides this, a lot of work regarding electronic theorization, building and testing is realized but not documented on this memory. This is a really practical project and most of the parts of the process can't and shouldn't be explained on paper.

Contents

1. Glossary

ARM is a microcontroller architecture, widely extended for its low cost and power consumption.

A **photointerrupter** is a device formed by a photoemitter and a photodiode that is able to capture movement using a light signal.

An **encoder** is a sensing device that transforms information from a format to another.

An **Integrated Development Environment (IDE)** is a computer application that facilitates the development and programming of software.

PWM is the pulse-width modulation of a signal in order to transmit information or control the energy provided.

An **Analog to Digital Converter (ADC**) is a tool that converts an analog signal to digital information.

A **Digital to Analog Converter (DAC)** is a tool that converts digital data into an analog electric signal.

An **Integrated Floating Point Unit (FPU**) is a part of a microcontroller or computer designed to carry out operations on floating point numbers.

A **debugger or debugging tool** is a computer program that is used to test and debug developed software into the controlling devices.

A **shunt** is a low-impedance resistor that is used to measure current.

A **digital buffer (or a voltage buffer)** is an electronic circuit element that is used to isolate the input from the output, providing either no voltage or a voltage that is same as the input voltage.

2. Preface

2.1. Origin of the project

This project is, together with the one carried by Carlos Conejo [1], the direct continuation of the “Two wheeled vehicle robots” line of investigation developed in the IOC (Institute of Industrial and Control Engineering). The general aim of all this research is to control and simulate vehicle’s behavior on a real road, being the first major step the recreation of a platooning system.

In the first three projects, the basics needed to achieve the final objectives were more than notably established. While Iván Prats assembled, modeled and programmed the first concept of the robot in his thesis [2], Antoni Riera created a Wi-Fi communication system in order to command and exchange data with the vehicles using a computer [3]. Finally, Albert Costa [4] joined the previous work and recreated some simulations, using the results to improve the robot’s controllers.

Although after all this work a functional and well behaving vehicle was created, during its development some hardware flaws were noticed, and here is where this task begins. This project tries to improve the robot implementation, adding some new electronic components and circuits. These upgrades also allow for renewing the motors control, thus improving the vehicle stability and response.

2.2. Motivation

When deciding the theme of the final degree thesis, the project exchange of the school was used in order to get access to a huge amount of ideas published by the same teachers that would tutor the research. Among all of them, the title and description of this project drew my attention and it quickly highlighted above the rest. After a quick interview with the project’s tutor, the decision was easily made. The explanation of what will the project consist completely fitted my interests and skills and the fact that I already had some experience in electronic welding and design with my high school research project was also a key factor. Finally, the possibility to work at the IOC with a group of experts in the field, getting access to their wise help and experience, was an opportunity that I could not pass by.

3. Introduction

3.1. Objectives

The main objective of this project is to assemble a two-wheeled vehicle robot and control its motors using the current as the main variable. In order to achieve the primary goal, some minor objectives have to be realized, such as:

- Use reverse engineering to obtain the circuit schematics of the original robot developed on Iván's project [2]. Analyze the circuits and understand all the electronics.

- Assemble a new robot using the schematics and all the components needed, while making some minor modifications to improve it.

- Develop a current sensor that reads the current of each one of the motors of the vehicle and sends it to the microcontroller. Design and add to the rest of the electronics the circuit that implements it.

- Model the motors and develop the control of their wheels using the current that flows through them. Design and calculate the parameters of the controllers that manage them.

- Implement all the control of the motor using the STM32f4-Discovery board as the microcontroller device. Learn C programming language and use it to program the tool.

Achieving all these objectives require a constant learning in order to understand and be able to realize all the tasks required. Skills such as electronic welding, oscilloscope manipulation and programming are constantly put to the test along all the process.

3.2. **Starting point**

Just some weeks before the beginning of this task, the third project regarding this subject was presented and evaluated. Three different works that once joined together create a really complete robot with some useful and interesting features.

The two-wheeled vehicle is mathematically modeled and a trajectory control is developed and implemented in order to make it follow a desired path. Using some sensors and the microcontroller, the vehicle is able to smoothly follow a black line and stop when detecting a front obstacle. This feature is not modified on this project and is used just as it is right now, as it is considered to be work correctly.

In addition, the vehicle motors are also modeled and controlled using a voltage control. However, the method used has some flaws as using the voltage to control the motors is not really efficient. This is where most of this project is based, completely renewing the motors control and using the current instead of the voltage as the main control variable.

Finally, the robot also has a completely original communication system using a Wi-Fi module. The vehicle is able to communicate with a computer, exchanging information and using a python application to modify control parameters in real time. On this project this communication system is enabled but it is not used or modified at all. Because of this, this feature is not explained on the memory and it is needed to read Antoni Riera's thesis [3] to understand it.

4. Hardware implementation

One of the major objectives of this project was to build another two-wheeled vehicle and implement some modifications and improvements to its hardware. Although Iván Prats [2] made some descriptions of all the robot's components, the electronic schematics of its implementation were never properly documented. Because of this, it was needed to check the already made robot in order to do reverse engineering for understanding and reproducing all the electronic circuits.

In this section, the mentioned description and documentation of the components and their circuits will be made, both for the already existing in the previous robot and for the new ones added in this project. All the schematics were made with Circuit Maker, a free program by Altium that allows you to create online designs using public libraries, in addition to drawing their respective PCB patterns.

4.1. List of components

The hardware used in the building of the robot were the following:

- 1 STM32F4-Discovery board.
- 1 two-wheeled vehicle chassis, including 2 DC bidirectional motors with its wheels and rotative encoders.
- 1 motor driver L298N.
- 1 ultrasonic sensor HC-SR04.
- 1 line and light sensor LRE-F22.
- 2 photointerrupters RPI-574.
- 1 WiFi module ESP8266
- 1 regulator LM317.
- 1 operational amplifier.
- 1 SN74HC04n integrated circuit with 4 inverters.
- 1 battery holder for 4 AA batteries.
- Some resistances, capacitors, diodes and other common electronic

To obtain some of this components, the Arduino kit “Kit Robot LRE-EO2” (*Fig.4-1*) was bought from www.leantec.es (a Spanish electronic shop), without the included Arduino UNO board. This pack contained the chassis, the motor driver, the photointerrupters and the ultrasonic and line sensors. It also features an USB cable to write the software into the board, the battery holder and some colour cables with a small protoboard for simple testing.

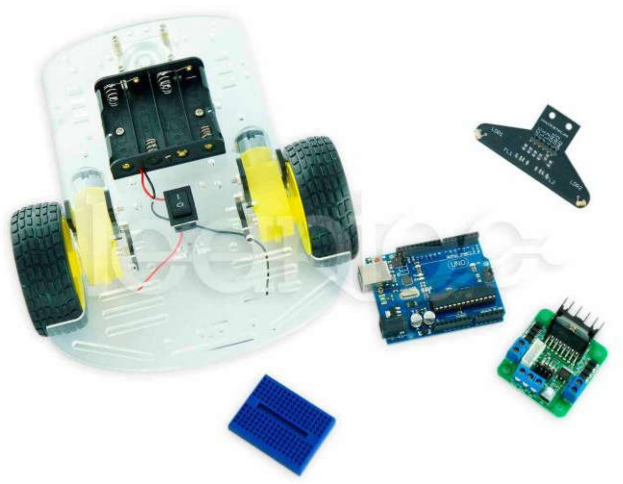


Fig. 4-1: Arduino kit “Kit Robot LRE-E02”

The Discovery board was acquired from STMicroelectronics instead of using the Arduino that could be included in the kit, and the Wi-Fi module was also bought on the same page as the vehicle pack. The rest of the elements were taken from the IOC lab, which has an extensive inventory of common electronic components and integrated circuits.

In addition to all this, some tools and devices were required for the building of the vehicle, such as a soldering iron for welding all the circuits and components. It was also indispensable to have access to an oscilloscope, a power supply unit and a multimeter in order to conduct all the tests and measurements.

4.2. General structure

On *Fig.4-2* the general structure of the electronic circuits of the vehicle is shown. It is centered around the Discovery board, with all the different sub-circuits and components connected to the device, either for controlling it or for converting their analog signal to digital information.

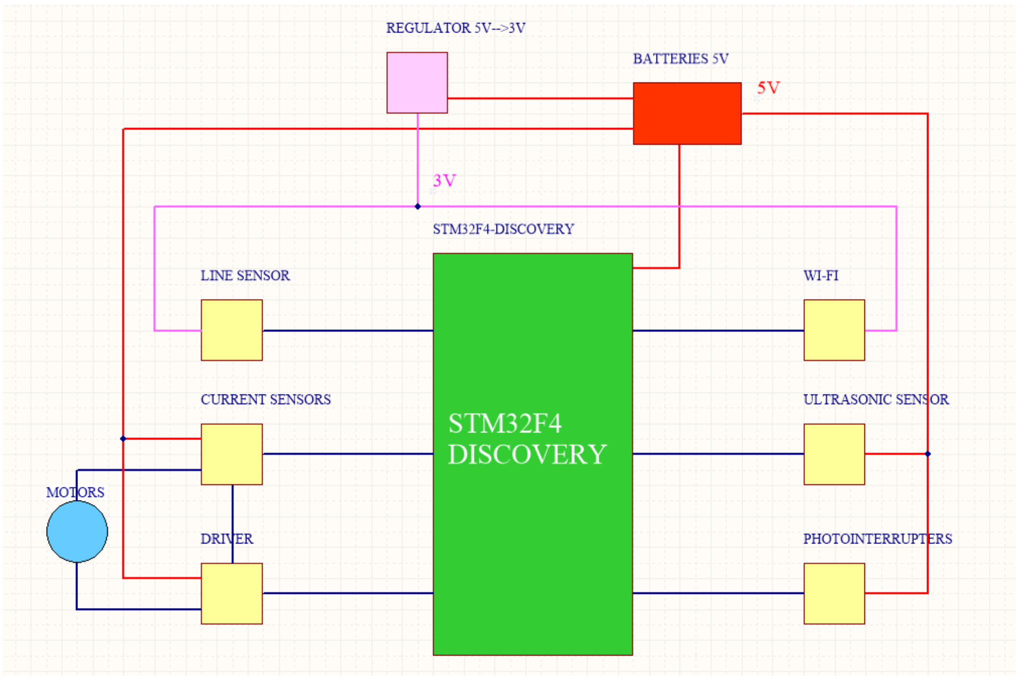


Fig. 4-2. General structure of the electronic circuits

Most of the components are powered by the 5V provided by the batteries, with the exception of the line sensor and the Wi-Fi module, which need a 3V input. For this purpose, a simple regulator circuit is implemented for converting the mentioned 5V signal into a 3V supply.

Note the layout of the current sensors, one of the main additions of this project over the previous. They are placed between the driver and the motors in order to measure the current intensity flowing through them. In later chapters their design and implementation will be described in depth.

4.3. STM32F4 Discovery Board

Controlling the motors and both acquiring and manipulating all the signals of the robot sensor is a demanding task, which requires a thorough microcontroller. The Arduino UNO board included in the previously mentioned kit could have been used, but it was found a device that better suited for this project.

The STM32F4-Discovery (*Fig.4-3*) is a board developed by STMicroelectronics that includes an STM32F407VG microcontroller based on ARM-Cortex architecture, along with all the components needed to make it function properly and some useful peripherals like timers, PWM channels and ADC/DAC converters. It is widely known as a cheap and accessible device, which makes it one of the most common microcontroller boards of the market. The main reason for this easy access is the availability of many freeware tools that allow for configuring and programming it without having strong programming skills. It is also considered a high performance microcontroller, with an Integrated Floating Point Unit (FPU) that allows really fast calculations.

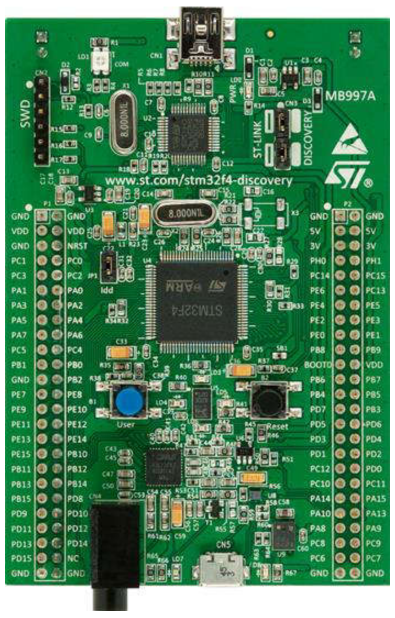


Fig. 4-3. STM32f4-discovery board

4.3.1. Programming software

As said above, the Discovery board presents a good amount of available software to facilitate its programming, and here it is going to be described which tools were used and their function. All the installing process was made following a series of tutorials founded on an online electronics blog [5].

The controlling program was developed in *C* language using *Eclipse*, a popular free and Open Source IDE (Integrated Development Environment). It was needed to install the *GNU Arm plug-ins* for *Eclipse*, which allows for developing and running codes for ARM microcontrollers such as the STM32 platform. Finally, the *GCC ARM tool-chain* was also added for compiling the program into the low level language. Once the firmware is developed and compiled, it is needed to debug it and send it into the microcontroller. For this purpose it was used the *OpenOCD debugger*, a universal free On-Chip-Debugger able to interface the most common platforms, including ARM-Cortex processors like the STM32. It was also needed to setup *Eclipse* for recognizing and using the OpenOCD tool.

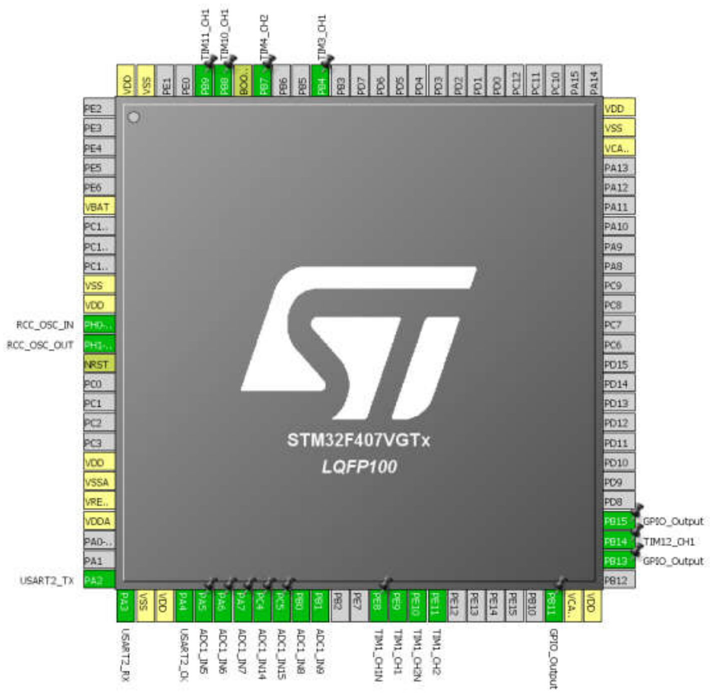


Fig. 4-4: STM32 Cube interface with the peripherals and pins used

Finally, there were used two other freewares that made the task of programming all the peripherals much easier: the *Hal libraries* and the *STM32 Cube.* The libraries contain all the code needed to habilitate and configure the peripherals and provide an immense amount of configuration possibilities. On the other hand, the *STM32 Cube* offers an easy to use interface (Fig.4-4) that allows for choosing the peripherals and their configuration and for automatically generating the code for copying it.

4.3.2. Circuit schematic

Fig. 4-5 shows the implementation of the STM32-Discovery board, along with the pins that are used for acquiring or controlling the signals of the peripherals. The labels represent connections with the other circuits of the robot and are named as their respective board pin in order to make the identification easier on other schematics.

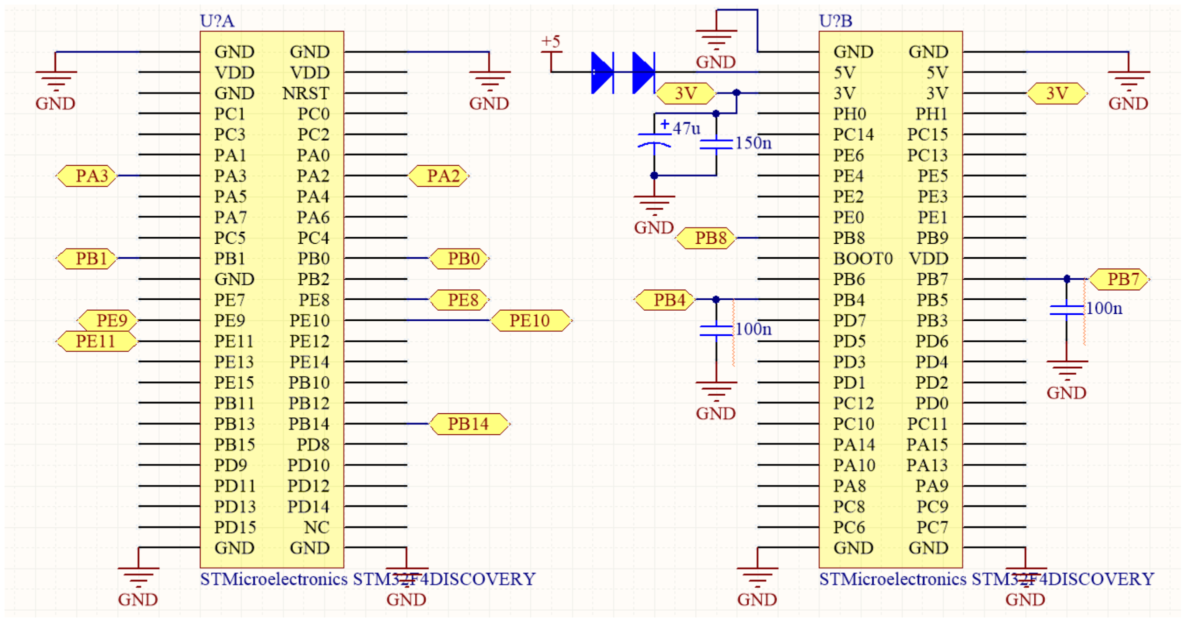


Fig. 4-5: STM32F4-Discovery circuit schematic

The board is powered with 5V, as said on the general structure section. Two diodes are placed on the input for protect the microcontroller of any kind of overvoltage. This kind of security measure can be seen along all the robot circuits, but it is especially important here in order to allow for working with the board connected to the computer without damaging it.

The STM32 also has an internal converter that allows it to offer an external 3V supply. It could be used to power the components that need a 3V input like the Wi-Fi module, but an external converter is implemented to avoid any problems. However, it is used on voltage security limiters (like the commented above) for other circuits. Given this, a pair of capacitors are connected for decoupling reasons.

The rest of the pins connected and their use are the following:

- PA2 and PA3 are connected to the Wi-Fi module and manage the exchange of information via PWM.

- PB0 and PB1 are enabled as a pair of channels of the ADC1. They convert the analog signal of the line sensor into digital information.

- PE8, PE9, PE10 and PE11 send the controlling signals to the motor driver. PE8 and PE9 correspond to the left motor control, being one the complementary of the other, with PE10 and PE11 behaving the same way but for the right motor.

- PB8 and PB14 trigger and acquire the signal of the ultrasonic sensor.

- PB4 and PB7 read the right and left photointerrupter measure respectively, to allow the calculation of their frequency. Note that they also have decoupling capacitors connected.

4.3.3. Used peripherals

The STM32 presents a large amount of peripherals that make it such a useful microcontroller. From all the tools available, the ones used on the project and their function are:

- 1 Advanced Control Timer (TIM1). It is the used for generating the signals of the motor driver, controlling the DC motors. It also triggers the main interruptive routine, which contains the current control.

- 2 General-purpose Timers (TIM3 and TIM4). They are configured in Input Capture direct mode in order to read the frequencies of the signals of the photointerrupters, thus calculating the speed of the wheels.

- Another General-purpose Timer (TIM2). It is used for triggering the secondary interruptive routine that contains the trajectory control and the speed control.

- 2 more General-purpose Timers (TIM10 and TIM12). They are in charge of triggering the ultrasonic sensor and then acquiring its response.

- 1 Analog to Digital Converter (ADC1) with 4 input channels. Two of them are used for reading the signal of the line sensor. The other pair is responsible for acquiring the values of the current sensors. All the acquisitions of the ADC are also made through a Direct Access Memory (DMA).

- 1 Digital to Analog Converter (DAC). It is used for sending digital information to the oscilloscope in order to graph and analyze its behavior.

4.4. LM317 Regulator

As said in previous sections, it is needed a 3V supply to power both the Wi-Fi module and the line sensor. For this purpose it is used a LM317T regulator, which has the pin distribution and recommended circuit of *Fig.4-6*.

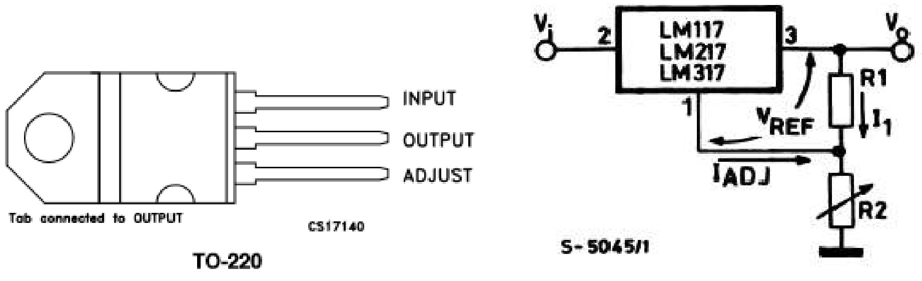


Fig. 4-6: LM31T pin distribution and circuit

In order to choose the value of R1 and R2 it is used equation 4.1, which is given by the datasheet with the value of $V_{REF} = 1{,}25V$.

$$V_o = V_{REF}\left(1 + \frac{R_2}{R_1}\right) + I_{ADJ}R_2 \tag{4.1}$$

Knowing this and the fact that I_{ADJ} can be up to 100µA,it is possible to neglect I_{ADJ} if it is chosen a small enough R2. With R1=1,5kΩand R2=2,2kΩ, it results in Vo=3,2V, which is considered a correct value for powering the devices. With these specifications and a pair of capacitors for decoupling, the circuit schematic is showed on *Fig. 4-7.*

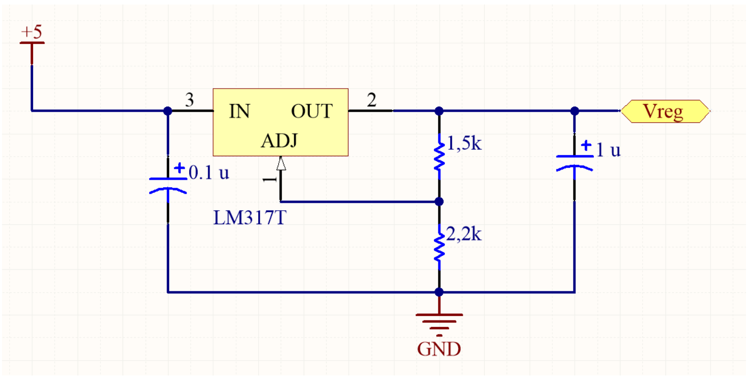


Fig. 4-7:. LM317 Regulator circuit schematic

4.5. Encoders and photointerrupters

For capturing and measuring the speed of the wheels, the encoders and RPI-574 photointerrupters that come with the vehicle kit are used. The photointerrupters work with two complementary elements: a LED that emits a signal and a photodiode that captions it. The encoders are attached to each of the vehicle wheels, rotating at their respective speed and placed between the two mentioned elements. This way, the response of the photointerrupter is interrupted at a pace proportional to the wheel speed, allowing measuring it by reading the frequency of the signal. For implementing the photointerrupters circuits it was followed the suggested scheme of their datasheet and it were added some complements to treat the signal, as shown on *Fig. 4-8*. The resistances values are chosen for maximizing the precision.

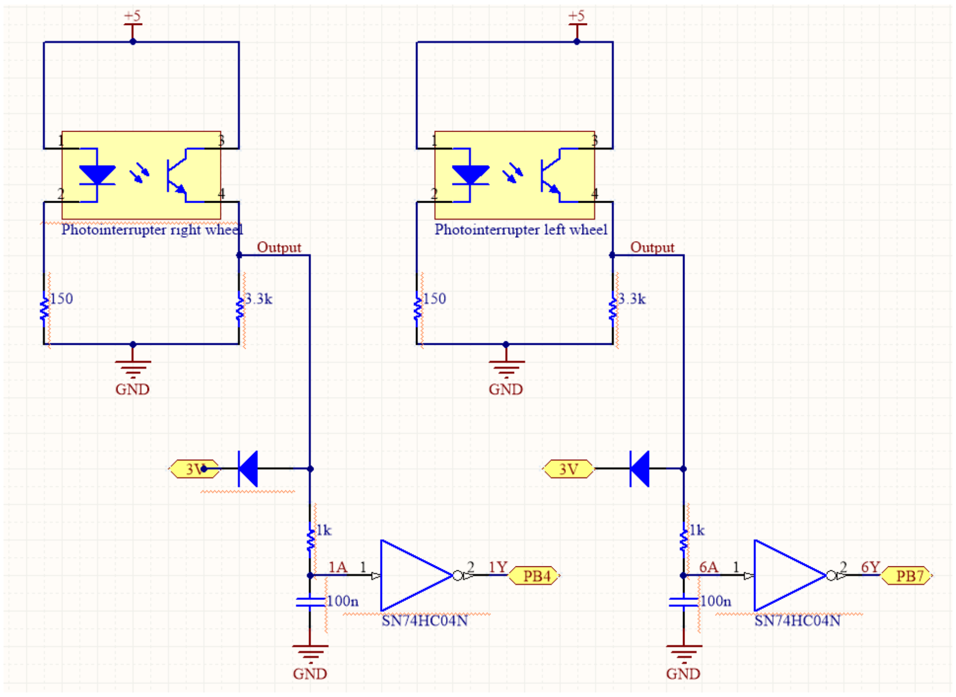


Fig. 4-8: Photointerrupters circuit schematic

The diodes are connected to the 3V supply of the STM32, thus creating a security voltage limiter like the previous seen. A 1st order low-pass filter is also implemented on both circuits in order to clean the signal to allow the microcontroller to read it correctly. Finally, an inverter is used as a digital buffer for removing disturbances. Both inverters used come from an SN74HC04N integrated circuit, which contains 4 of them and needs to be powered with a 5V input.

4.6. HC-SR04 Ultrasonic sensor

An ultrasonic sensor is needed for detecting obstacles in front of the vehicle and the HC-SRO4 *(Fig. 4-9)* that come with the vehicle kit bought is used for this purpose. In this project the obstacle detection does not affect the robot control and it is only useful for stopping the vehicle to avoid collisions. However, in future works that allow for multiple vehicles flowing together in a platooning system this sensor will be used for controlling the distance between them.

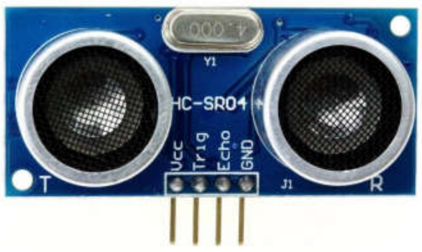


Fig. 4-9. HC-SRO4 Ultrasonic sensor

The sensor performance is based around measuring the travelling time of an ultrasonic echo *(Fig. 4-10)*. First of all, the microcontroller triggers the device by sending a 10µs pulse and the sensor emits 8 ultrasonic burst that will bounce back once they find an obstacle. As soon as the final burst is send, the echo signal goes to high level until the pulse is received back or a new trigger is sent.

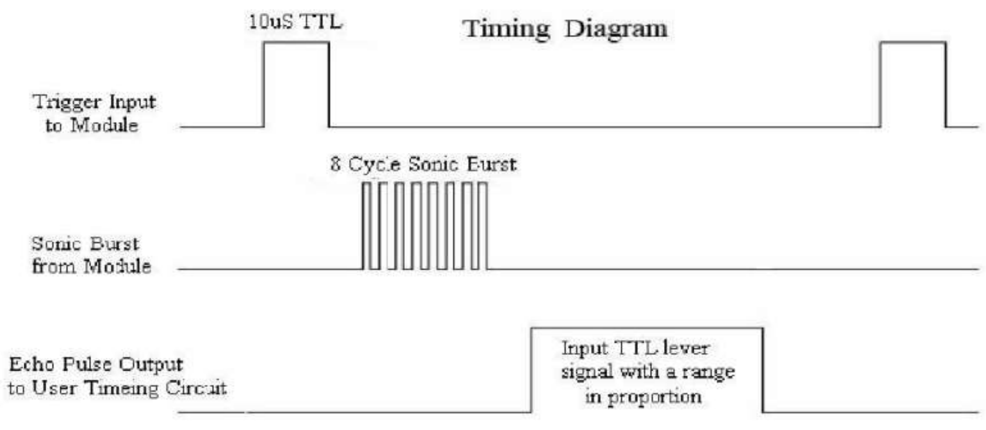

10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

Fig. 4-10: Timing diagram of the HC-SR04

The microcontroller receives de echo signal and measuring the period that it is on high level it calculates the distance between the vehicle and the obstacle, using a simple proportion:

$$distance[cm] = t[\mu s] * 0{,}01715 \qquad (4.2)$$

The circuit schematic *(Fig 4-11)* is really simple, as it does not need any components outside of the direct connections. The sensor is powered with the 5V batteries supply and the trigger and echo channels are connected to the Discovery board by pins PB8 and PB14 respectively.

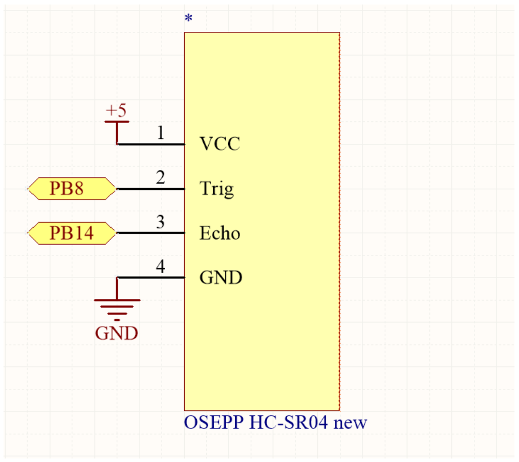


Fig. 4-11: Circuit schematic of the HC-SR04 implementation

4.7. LRE-F22 line sensor

The LRE-F22 infrared sensor of the vehicle kit is used for knowing if the robot is above the line or measure the distance between them. The device also presents a light sensor, but it is not used on this project.

The tool functions using a pair of infrared emitters and two receivers that capture the infrared light that reflects on the surface below it. Depending on the color of the surface the amount of light reflected will vary, with black surfaces absorbing it all and light surfaces doing the opposite. Then, two output signals are generated (FL1 and FL2, one for each receiver) that go from 0V to VCC depending on how much infrared light is received. Knowing this and using a black line to follow, it is possible to calculate the distance between the vehicle and the line using the value of this signals.

The electronic circuit that implements the sensor *(Fig 4-12)* is really simple and it uses some concepts that have been seen before on the project. First of all, it is powered with the 3V provided by the regulator, meaning that the output signals of the receivers go between 0V and 3V. Then this signals are sent to the STM32 through pins PB0 and PB1, being captured by two channels of the ADC1. Note that low-pass filters and voltage limiters are added as in some other sub-circuits of the robot.

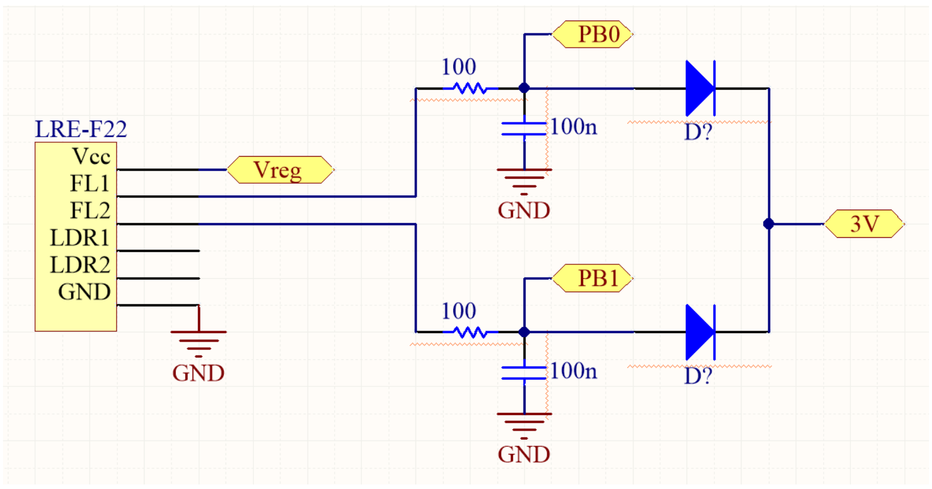
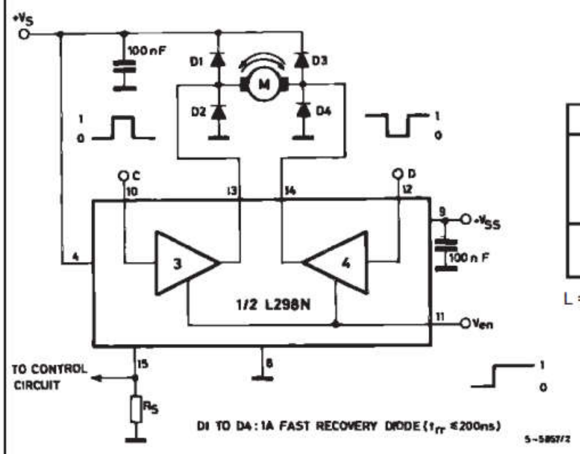


Fig. 4-12: Circuit schematic of the LRE-F22 line sensor implementation

4.8. **L298N Driver**

A motor driver is needed in order to control the speed and direction of the DC motors. In this project it used the L298N Driver that comes with the vehicle kit bought. This device contains a pair of H-bridges which allow to invert the direction of the currents, thus also inverting the direction of the motors rotation. It also contains 8 fast-recovery diodes for overvoltage protection, but the original components are low quality and generate disturbances on the response, so they are changed to better ones.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Fig. 4-13: Internal circuit of the and Input/Function table of the L298N driver

The driver controls each of the motors using a pair of complementary signals (C and D), which are provided externally by the microcontroller. As seen on the table of *Fig. 4-13*, depending on the power level of this signals the motors rotate on forward or reverse direction. The speed of the motors is determined by which percentage of a determined period the mentioned signals are on each level. This percentage is known as the duty cycle and is expressed as Δ[%], having one of them for each wheel.

With a duty of Δ=100%, C signal will be on high level all the time and D on low, meaning that the motor goes at full speed on forward direction. On the other hand, a duty of Δ=0% is the complete opposite and the motor rotates backwards at its maximum speed.

The motor is stopped with a duty of Δ≈50% and values close to the middle make the wheels move really slowly, while their speed is increased as the duty gets higher on each one of the directions.

The driver is controlled by the STM32 with 4 PWM signals, a pair for every motor (PE8 and PE9 for the left one and PE10 and PE11 for the right one), corresponding to the mentioned C and D inputs. The device is powered by the 5 V supply of the batteries and does not need any external security system, as it already contains the diodes seen before. Note at *Fig. 4-14* the R shunts placed between the driver and the motors. They are used on the current sensors, which are described on the following chapter.

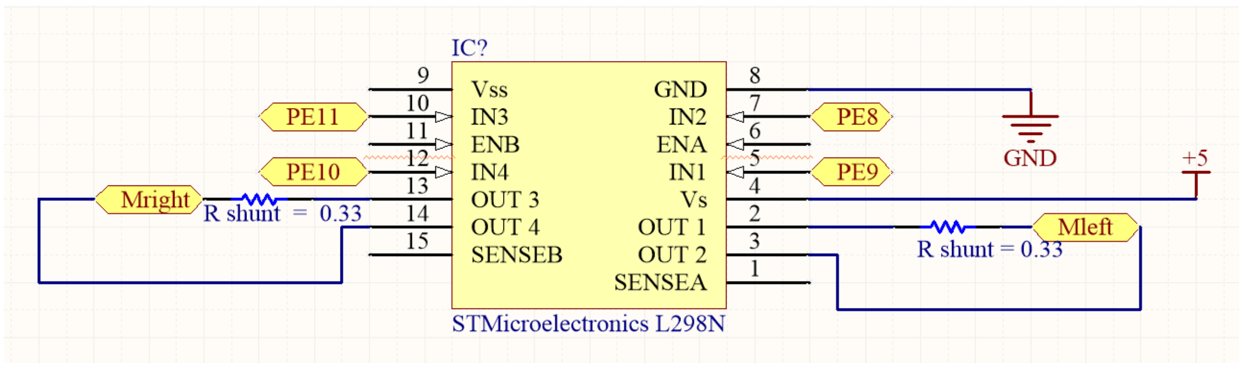


Fig. 4-14: Circuit schematic of the L298N driver

4.9. **Current sensors**

The main hardware addition of this project over the previous vehicle is the implementation of current sensors for each of the motors. With this addition it is possible to control the motors using a current control instead of a voltage one, being more precise and having a wider working speed range.

4.9.1. Design

The microcontroller reads the value of electric signals using ADCs, like it is made for the line sensor. However, it is only able to capture voltage values, so it is not possible to directly capture the current intensity and it is needed to use a shunt, a low-impedance resistor that allow the intensity to pass through and calculate it by measuring the voltage and applying the Ohm law (I=V/R) . The shunts used are placed between the DC motors and the driver, this way reading the current that passes through them.

For reading the voltage using the ADC its value is needed to be big enough or there would be too much errors, given the precision of the tool. Because of this, an amplification phase has to be implemented after the shunt. Another obstacle is that the ADC is not able to acquire negative values, so it is needed an offset. Doing this, the 0 V height is risen to the offset point and negative currents are the ones considered between it and the new 0.

Before choosing the value of the shunt resistances it is first needed to define the range and offset of the voltage acquired by the ADC. In this case, it is used a range of ±1.5V with an offset of 1.5 V. This way, positive currents are read as voltages that go from 1.5 to 3 V and negatives ones between 1.5 and 0 V.

Knowing this, it is intended to use resistors of 0.3 Ω with a current tolerance of ±0.5 A. Applying Ohm law the voltage of the shunts is ±0.150 V, which with an amplification gain of 10 and the mentioned 1.5 V offset it gives the desired values of the ADC acquisition.

4.9.2. Circuit schematic

Although on the design of the sensor there were chosen resistors of 0.3 Ω, in the lab there were not any resistances of this value so three 1 Ω resistors are connected in parallel, making an equivalent of ⅓ Ω as seen on *Fig 4-15*. Additionally, the amplification phase has to be adjusted to get a 9,1 gain for maintaining the same voltage range.

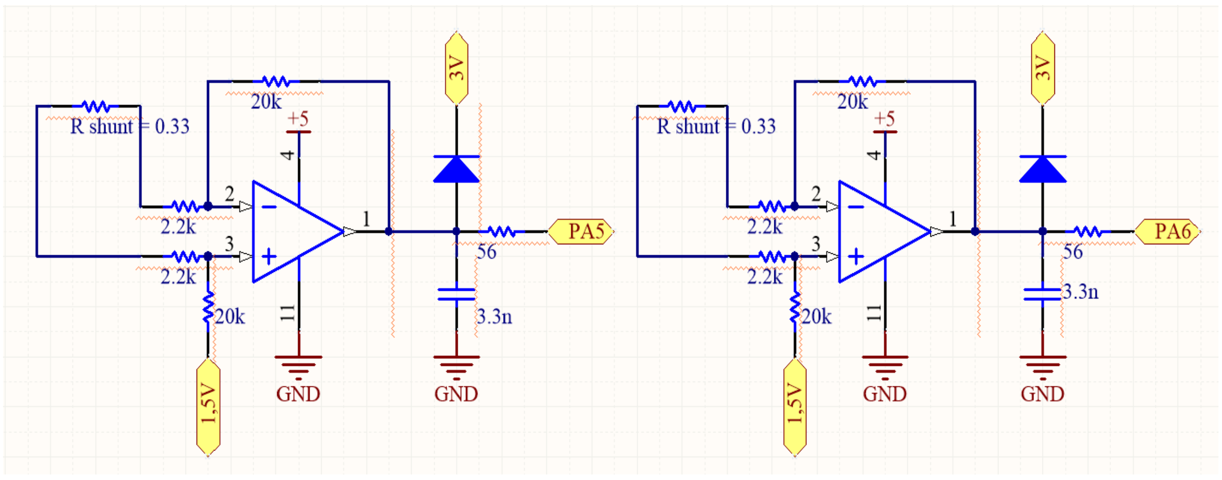


Fig. 4-15: Circuit schematic of the current sensors implementation

For the amplification phase there are used two operational amplifiers, both taken from the same integrated circuit and powered with the 5 V supply of the batteries. With this configuration and the chosen values for the resistances, the gain of the phase is

$$\frac{R2}{R1} = \frac{20k}{2,2k} = 9,09 \qquad (4.3)$$

which is almost the same as the desired value and transforms the shunt read into a signal of ±1.5 V. Additionally, instead of linking the positive input of the amplifiers with the ground it is connected to a reference signal of 1.5 V, providing the offset needed and modifying the output to the final 1.5±1.5 V.

Finally, the signal is treated before the ADC acquisition, just like it has been made with other circuits of the robot: a low-pass filter is implemented for removing high-frequency disturbances and a 3 V limiter is created for preventing overvoltage.

4.9.3. Reference signal

For providing the mentioned offset to the output of the current sensor it is needed to obtain a 1.5 V signal from some supply. It could have been used a battery of 1.5 V, but instead a voltage divider *(Fig.4-16)* is implemented, benefitting from having various 3 V sources.

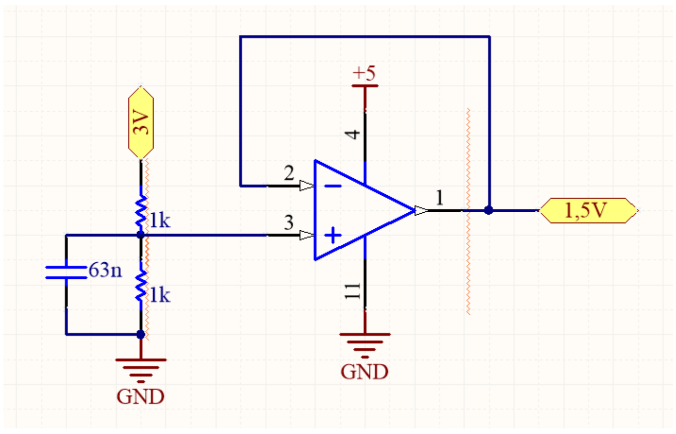


Fig. 4-16: Circuit schematic of the voltage divider that provides the reference signal

Two equal 1k Ω resistances are connected in series between the 3 V output of the STM32 and the ground, creating a voltage divider and providing 1.5 V on their middle point. However, it can't be directly used as the reference signal because the charge of the current sensor would act as another resistor affecting the voltage distribution and there would not be the desired 1.5V.

For avoiding this phenomenon, a voltage follower is implemented using another operational amplifier of the integrated circuit with unitary gain. This acts as a buffer, providing enormous input impedance that does not disturb the voltage division and allows the reference signal to have the correct value.

5. Modeling and design of the control

The main purpose of implementing current sensors is to allow for controlling the vehicle using the relation between the current intensity and the motor response. All the previous projects have used a voltage control but it limits the accuracy of the motors response. The current control on the other hand acts directly to the torque of the motors, providing better accuracy and a wider speed range.

The overall control of the robot is composed of three sub-controls working in a feedback sequence (*Fig. 5-1*). First of all, the trajectory control uses the measurements of the line sensor to calculate the desired speed of each wheel in order to follow the path. Then, the speed control compares the feedback of the encoders with the reference speed and provides the current value that would correct the error. Finally, the mentioned current control works on another internal feedback that adjusts the input of the motor for matching the reference current intensity.

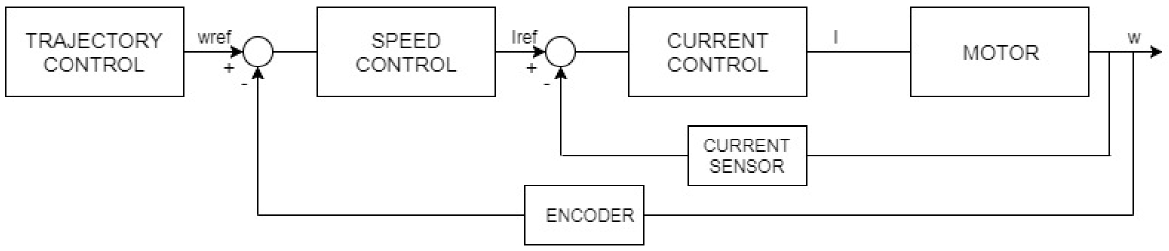


Fig. 5-1: General structure if the robot control

The control design carried out on this project is only based on the speed and current controls, as they are the ones affected by the change of using the current as the control variable. As for the trajectory control and the dynamic model of the vehicle that defines it, the design and calculations made on previous projects are used with no changes at all, so they are just briefly explained without going into depth detail.

5.1. Vehicle model and trajectory control

The two-wheeled vehicle was modelled in Iván Prats' thesis [2], which combined the ideas presented in two articles [6] [7] in order to develop a model that suits the robot's desired behavior. Using this model, he also designed the trajectory control that allows the robot to follow a desired path. In this section, this model and control are overviewed for better understanding of the general vehicle control, but it is recommended to check Iván's work for a more in depth explanation since none of the equations nor calculations are presented here.

The main concept used for designing the vehicle model and its control is the definition and parameterization of two elements: the desired path and the vehicle itself, having each one its respectful coordinate system (*Fig. 5-2*). This way, in order to ensure that the robot follows the line it is only needed to match the measure point (the line sensor location) and orientation of the vehicle with their trajectory respective.

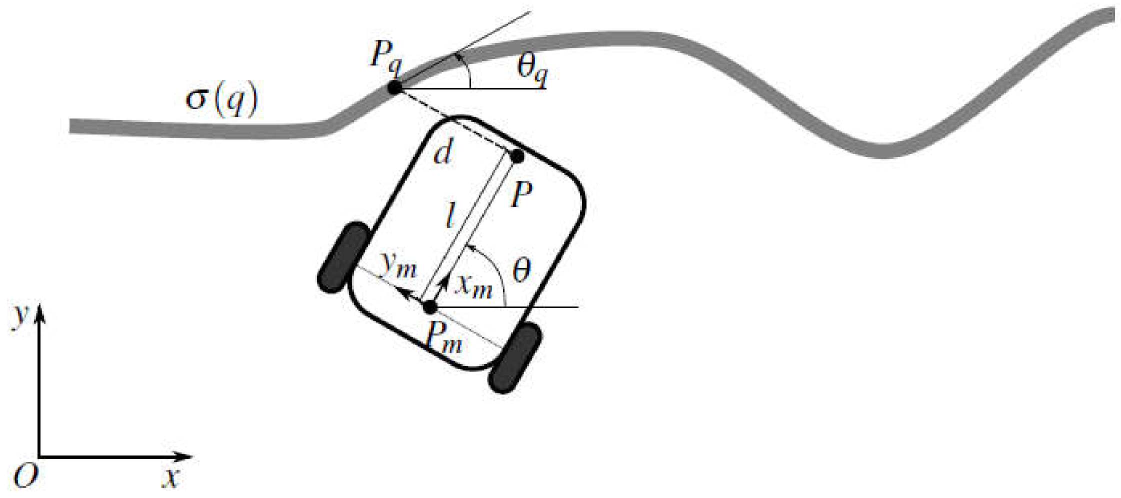


Fig. 5-2: Model of the vehicle and the trajectory. The vehicle is oriented with a local coordinate system (xm, ym), its measure point P and the angle θ. The trajectory σ(q) uses the global axes (x,y), the point Pq and the angle θq.

The kinematic model of the vehicle robot, on the point Pm, is given by the following equations systems:

$$\begin{cases} \dot{x} = u_1 \cos\theta \\ \dot{y} = u_1 \sin\theta \\ \dot{\theta} = -u_2 \end{cases} \qquad \begin{cases} u_1 = \frac{r}{R}(w_R + w_L) \\ u_2 = \frac{r}{2R}(w_R - w_L) \end{cases} \qquad (5.1)\ (5.2)$$

Where r is the radius of the two wheels, R the distance between them and w_R and w_L the angular velocity of the right and left wheels, respectively.

With the vehicle model defined, two conditions must be fulfilled in order to control the robot and make it follow the desired path:

-$P_q = P$, which is equivalent to say that d=0.

-$\theta_q = \theta$, guaranteeing that the robot is aligned with the trajectory.

The controller implemented is a Proportional Integral (PI) and its aim is to force d=0, keeping the vehicle over the line. To do this it is usedu_2, while u_1 is considered steady and both variables are kinematically related with the wheel's speeds as showed on equation " ". *Fig. 5-3* describes the process followed by the trajectory control.

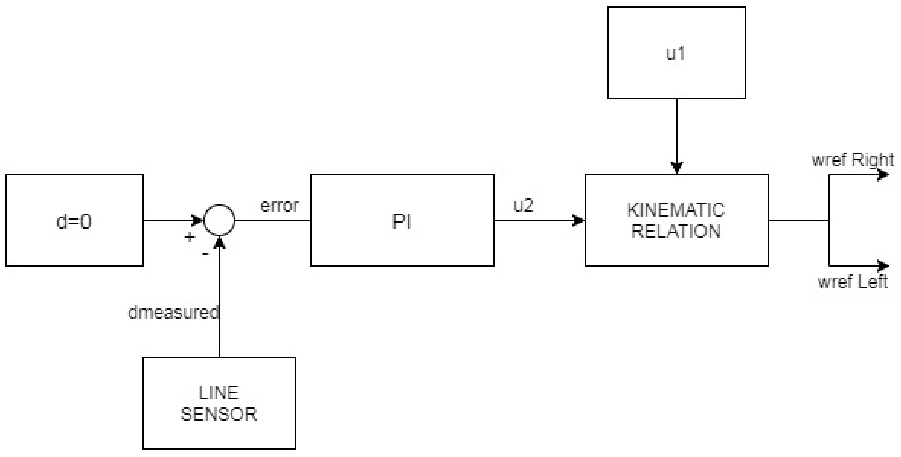


Fig. 5-3: Structure of the trajectory control

5.2. **Modelling of the DC motors**

The motors of the vehicle are modeled using the most common structure, which considers the motors to also have an armature resistance R_a and an inductanceL_a.

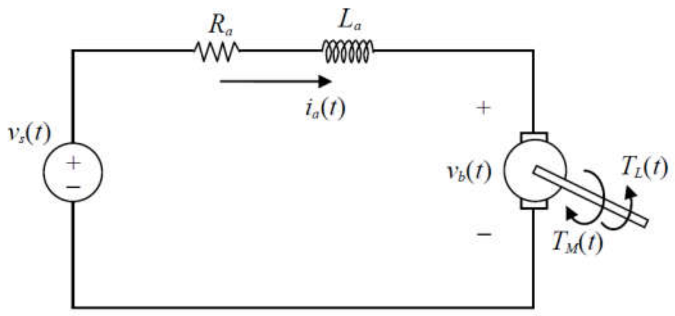


Fig. 5-4-Model of the motors

On *Fig.5-4* it can be seen the structure of the model, where v_s is the voltage source, v_b the emf (electromotive force voltage) and $i_a$the armature current. $T_M$is the torque of the motors and $T_L$is the external torque of the payload. With this model it is possible to describe the motors behavior using two equations, an electrical and a mechanical one.

Applying Kirchhoff's voltage law, the electrical equation that describes the motors is

$$L_a \frac{di_a}{dt} = -R_a i_a(t) + v_s(t) - v_b(t) \qquad (5.3)$$

Where v_b is proportional to the speed of the motors as

$$v_b = k_b w \qquad (5.4)$$

Some of the parameters of this model have been defined on the laboratory such as the resistance, which is directly measured as $R_a = 4{,}8\ \Omega$. Applying a constant angular speed to the motors without applying current using an electric screwdriver and measuringv_b, it has been possible to obtain $k_b = 0{,}156\ \frac{V*s}{rad}$.

On the other hand, the mechanical equation dependent of the angular speed is

$$J\frac{dw}{dt} = -Bw(t) + T_M(t) - T_L(t) \qquad (5.5)$$

With B as the frictional coefficient and J as the rotor moment of inertia. The motor torque T_M is proportional to the armature current, expressed as

$$T_M = k_t i_a \tag{5.6}$$

In this case it is only possible to characterize $k_t = k_b$, which has the same value calculated before. The other two parameters of the equation are not possible to obtain as it has been made with equation 5.3.

Finally, the motors behavior both mechanical and electrical can be described with the following equation system

$$\begin{cases} L_a \frac{di_a}{dt} = -R_a i_a(t) + v_s(t) - k_b w(t) \\ J \frac{dw}{dt} = -Bw(t) + k_t i_a(t) - T_L(t) \end{cases} \tag{5.7}$$

For the voltage control used on the previous work the two equations must be combined in order to obtain a relation between the motors speed and voltage. However, as in this project the control is made over the current it is only needed the mechanical one. Unfortunately, as it is not possible to know its parameters, in order to design the controller on later chapters the motor needs to be defined using its dynamic response.

5.3. **Current control design**

The current control is on charge of guaranteeing that the input current of the motors is equivalent to the desired value received by the speed control. As seen on *Fig. 5-1*, the current and speed control are strongly related between them. However, in order to design each one of them it is considered that the other one is perfectly working so it is not needed to take it on account. Therefore, the input Iref is assumed to be the proper one and all the experiments and comprobations are made with an arbritrary current. *Fig. 5-5* describes this control routine.

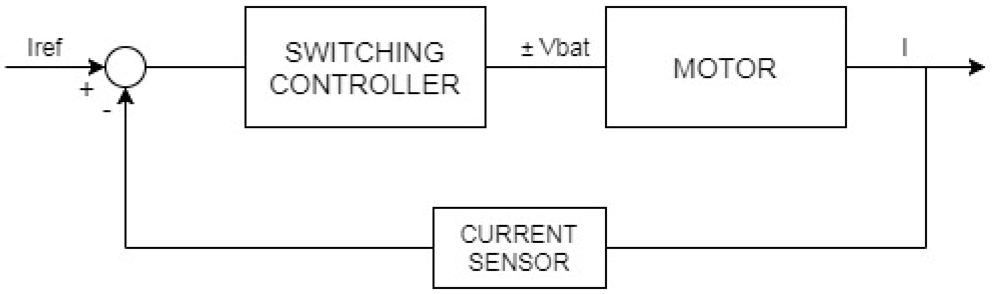


Fig. 5-5: Structure of the current control

5.3.1. **Switching controller**

The controller used for the current is a simple version of discontinuous nonlinear control: the switching controller. This type of control does not use a continuous fuction like the PI controller, instead it jumps from one condition to the opposite one at a very high speed. In this case, what it does is send to the motor the maximum value of voltage available from the batteries, switching the sign of it depending on if the real current is higher or lower than the reference one.

$$I_{measured} - I_{ref} \begin{cases} > 0, \ \ V = -V_{bat} \\ < 0, \ \ V = +V_{bat} \end{cases} \tag{5.8}$$

By doing this, the controller forces the real current to move towards the desired one and to keep bouncing up and down from the reference level, keeping its mean value as the correct current intensity, as shown on *Fig. 5-6*.

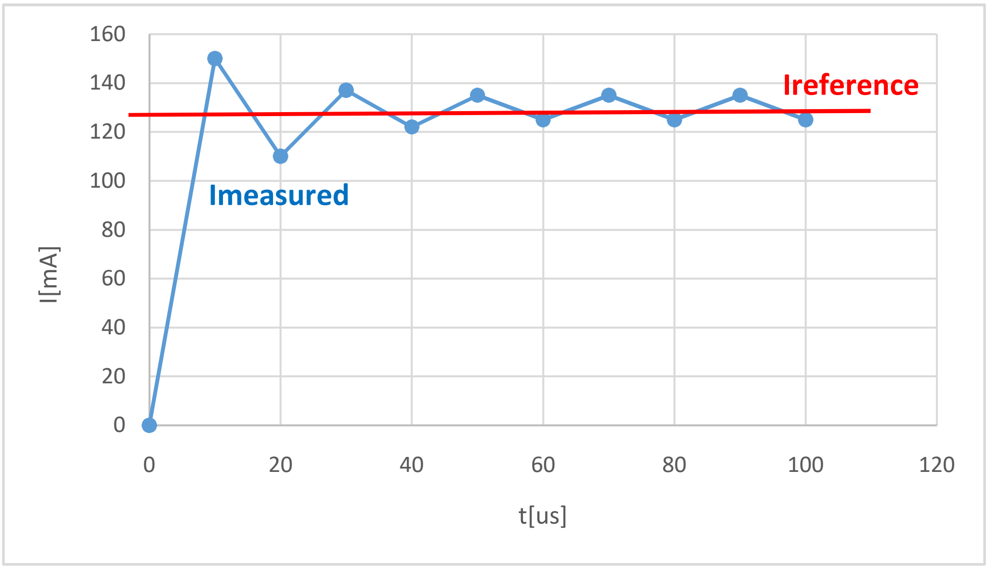


Fig. 5-6: Example of the switching controller behavior.

This type of control needs to be executed at a really high frequency or the current would not change its sign quick enough to stay centered on the desired value. For this reason, this routine is performed every 10µs, while the rest of the controls and calculations of the microcontroller are made with a 1ms period.

5.3.2. Experimental results

In order to analyze the response of the control and to check if it really works properly, some visual tests have been made using an oscilloscope. All the verifications have been made with the speed control disconnected, as the current control has to be analyzed separately as explained before.

To do this, the reference current has been forced to be a desired value using the microcontroller. With this process it is possible to program the current to change its value periodically, allowing displaying the controller behavior when exposed to steep current variations.

At *Fig. 5-7* it is showed the response of the current control when it receives an input that goes from -100 mA to 100 mA every half second. This is a really good test to check the controller performance in one of the worst scenarios possible, as not only it needs to execute sharp jumps but also change the direction of the current.

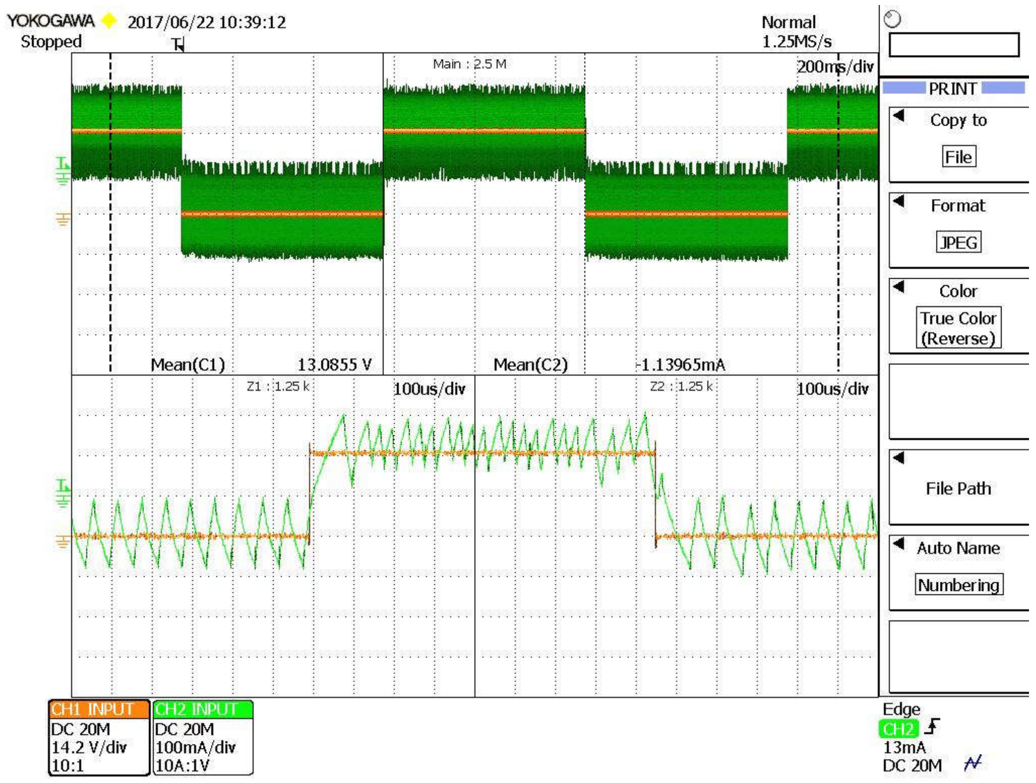


Fig. 5-7: Response of the current control to a -100mA/100mA input. The green plot is the current and the orange one is the DAC proportional to the reference current.

Looking at it, it is obvious that the control is performing quite well. The current is centered on the desired value at every stage, jumping from -100 mA to 100 mA just like the reference. Note the sharp tooth-tooth shape of the current signal, caused by the switching controller.

In respect of the swiftness of the response, it requires less than 100µs to change the current value and almost the same amount of time to stabilize again. Knowing that the rest of the robot controls are made every 1ms and that the controller won't be demanded on tasks as hard as this test, it is right to say that the current control will perform properly on every situation required by the vehicle's movement.

5.4. Speed control design

The control designed to make the motors rotate at the desired speed is based around two elements: a PI (proportional-integral) controller and a feed-forward relation between current and speed. The PI is on charge of calculating the error and applying the correction on the speed needed to minimize it. On the other hand, as the current/speed relation is not linear the feed-forward modifies the PI output to resemble the reality and to arrive to the steady state much quicker.

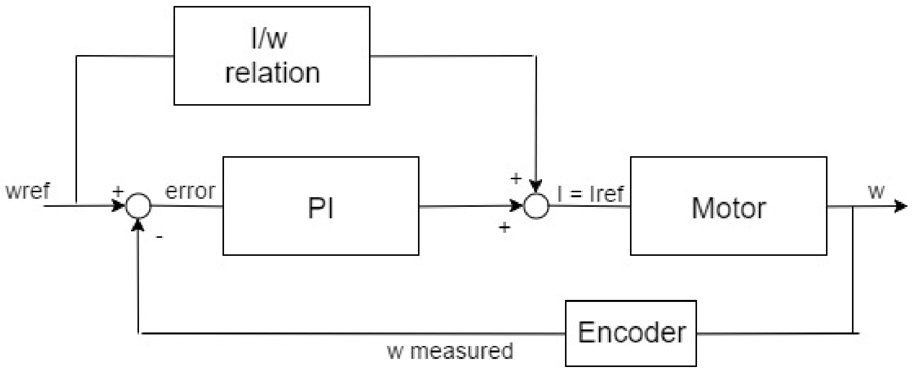


Fig. 5-8: Structure of the speed control

As commented before, the design of the current and speed control are made separately and considering the perfect behavior of the other one. Because of this, the input current I of the motor is assumed to be the desired Iref as if the current control had already corrected it.

On this section it is explained the characterization of the motors transfer function, the measurement and calculation of the feed-forward relation and the design of the PI controller.

5.4.1. Determination of the motor transfer function

In order to design the PI controller it is needed to consider the motor as the plant of the system, so it is needed to determine the transfer function that defines it. Looking at the diagram of *Fig. 5-8*, it is obvious that this function has a current input and a speed output, so it is used the mechanical equation of (5.8).

If the external torque $T_L$is considered neglible, applying Laplace transform the frequency equation in terms of s is

$$(Js + B)\Omega(s) = k_t I(s) \tag{5.9}$$

and by isolating the product w/i_a it is obtained the 1st order transfer function of the motors

$$\frac{\Omega(s)}{I(s)} = \frac{k_t}{Js+B} \tag{5.10}$$

With this function it would be simple to design a PI controller, converting the system into a 2nd order one and defining its response. However, as said before it is nearly impossible to determine the parameters k_t, $J$and$B$, so this function can't be directly used. Instead, the system is considered to have a generic 1st order transfer function like the following:

$$\frac{\Omega(s)}{I(s)} = \frac{K}{\tau s+1} \tag{5.11}$$

This transfer function has all the previous parameters implicit as $K = {}^{k_t}/_{B}$and $\tau = {}^{J}/_{B}$.

Now it is only needed to determine the plant's gain K and the time constant τ, which can be measured by plotting and analyzing the temporal response of the system.

This temporal response is plotted with the oscilloscope on *Fig.5-9* using the DAC to send the value of the encoder measurement and capturing the transitory behavior of the speed when a 150 A current is applied.To obtain the gain K, it is measured the voltage increase ΔV of the DAC, converted into the corresponding speed value and divided by the 0,150 A current. On the other hand, knowing that $4\tau = 98\% t_s$, dividing the duration of the transitory ΔT by 4 the time constant $\tau$is roughly calculated.

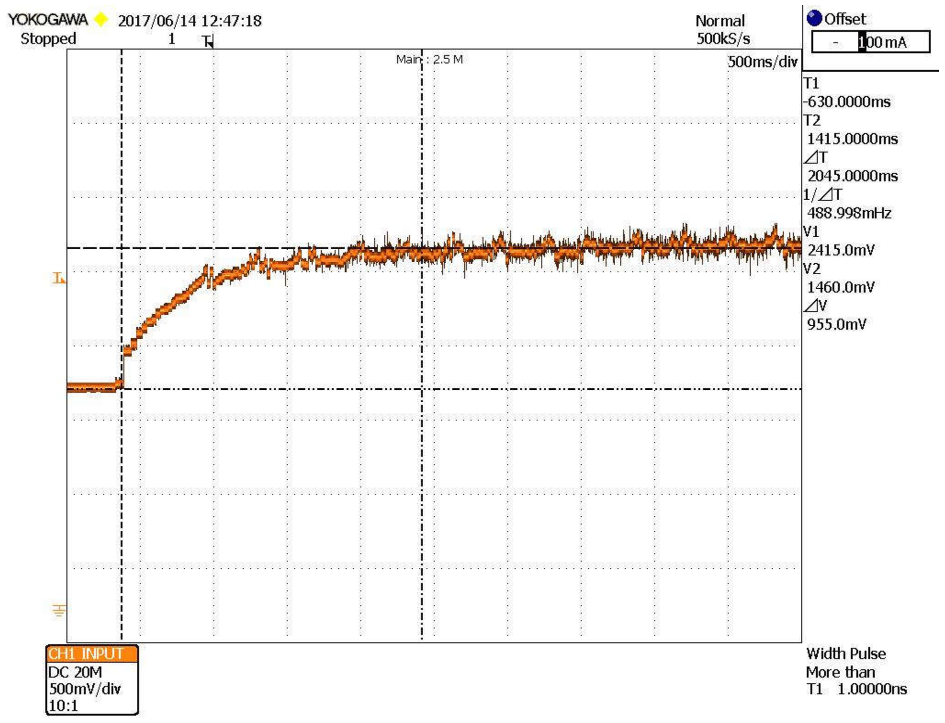


Fig. 5-9. Temporal response of the motor plant.

With the parameters defined, the final transfer function of the motor plant is

$$G(s) = \frac{\Omega(s)}{I(s)} = \frac{K}{\tau s+1} = \frac{133{,}7}{0{,}511s+1} \tag{5.12}$$

5.4.2. Design of the PI controller

The controller selected for managing the speed of the system is a proportional-integral one, as it is quite simple to design and allows for having a stable response with a low stationary error. This controller has a proportional part that minimizes the error at every moment and an integral one that considers the deviation over time. The function that defines the PI controller is

$$C(s) = k_p + \frac{k_i}{s} \tag{5.13}$$

The controller is applied directly before the motor plant, with the product between them becoming a 2nd order transfer function as

$$CG(s) = \frac{K(k_p + \frac{k_i}{s})}{\tau s + 1} = \frac{K(k_p s + k_i)}{\tau s^2 + s} \qquad (5.14)$$

Applying the negative feedback and considering the whole system, the general function that defines it is

$$W(s) = \frac{CG(s)}{1 + CG(s)} = \frac{K(k_p s + k_i)}{\tau s^2 + (Kk_p + 1)s + Kk_i} \qquad (5.15)$$

With the system's transfer function determined, it is now needed to apply some specifications in order to choose the desired $k_p$andk_i. In this case, the pole placement method is ideal, as the system has a 2nd order response and its poles can be determined just by defining a couple of parameters.

The two poles of the system have a general form like this

$$p_1, p_2 = -\sigma \pm jw_d \qquad (5.16)$$

With σ as the real part that defines the system's stability and w_d as the complex part that determines its oscillation. By definition, the poles of the system are the roots of the denominator of the system's function, so they can be defined as

$$D(s) = (s - p_1)(s - p_2) = (s + \sigma - jw_d)(s + \sigma + jw_d) \qquad (5.17)$$

$$= s^2 + 2\sigma s + (\sigma^2 + w_d^2) = 0$$

On the other hand, going back to the general transfer function the roots can also be expressed as

$$D(s) = 1 + CG(s) = \tau s^2 + (Kk_p + 1)s + Kk_i \qquad (5.18)$$

$$= s^2 + \frac{(Kk_p + 1)}{\tau}s + \frac{Kk_i}{\tau} = 0$$

Comparing the two equations it is obtained the PI constants in function of the system's poles

$$\begin{cases} k_p = \frac{2\sigma\tau - 1}{K} \\ k_i = \frac{(\sigma^2 + w_d^2)\tau}{K} \end{cases} \tag{5.19}$$

Additionally, the poles parameters are defined by the 2nd order response of the system, specifically by the establishment time t_s and the overweight $S_p$according to this relations

$$\sigma = \frac{4}{t_s} w_d = -\frac{\pi\sigma}{\ln{(S_p)}} \tag{5.20}$$

With all this, the PI constants $k_p$and $k_i$are completely defined and their values can be freely chosen just by modifying the temporal response of the system.

5.4.3. Feed-forward implementation

The feed-forward input is an empirical implementation that helps the controller to achieve the desired speed quicker and smoother. In order to apply it, it is needed to estimate a relation between the motor speed and the input current on all the speed range. With this relation, the current that goes after the PI is forced to move towards the value equivalent to the reference speed. This way, the next iteration of the control is much closer to the desired value and the number of steps required is minimum in comparison with the original control.

For obtaining the mentioned relation, the speed of the wheels was measured using the encoders when receiving known values of the input current, as showed on *Fig. 5-10.* Note that the motors don't move when the current is below ±0.079 A. With these measurements, the relation obtained is

$$I(w) = \begin{cases} 0.003w - 0.079, & w < 0 \\ 0.003w + 0.079, & w > 0 \end{cases} \tag{5.21}$$

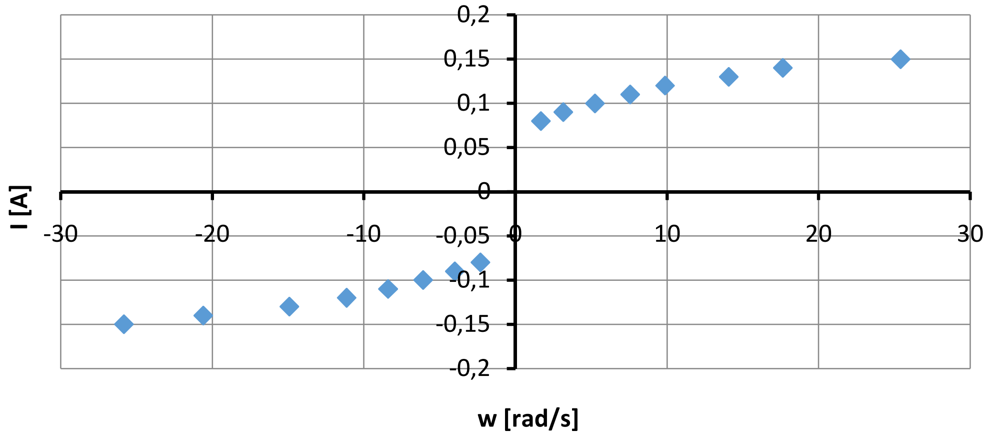


Fig. 5-10: Experimental relation between current and speed.

However, when applying and testing this relation it was discovered that it only worked when the vehicle was already moving and it didn't help to start rotating the wheels. After some tests, it was discovered that the motors need an initial input 0,250 A in order to overcome the resistant torque. This phenomenon can be seen on *Fig. 5-11,* which shows the response of the control when the motors start moving. This test was realized with a really slow PI controller, for better display of the results and the differences between this relation and the final one.

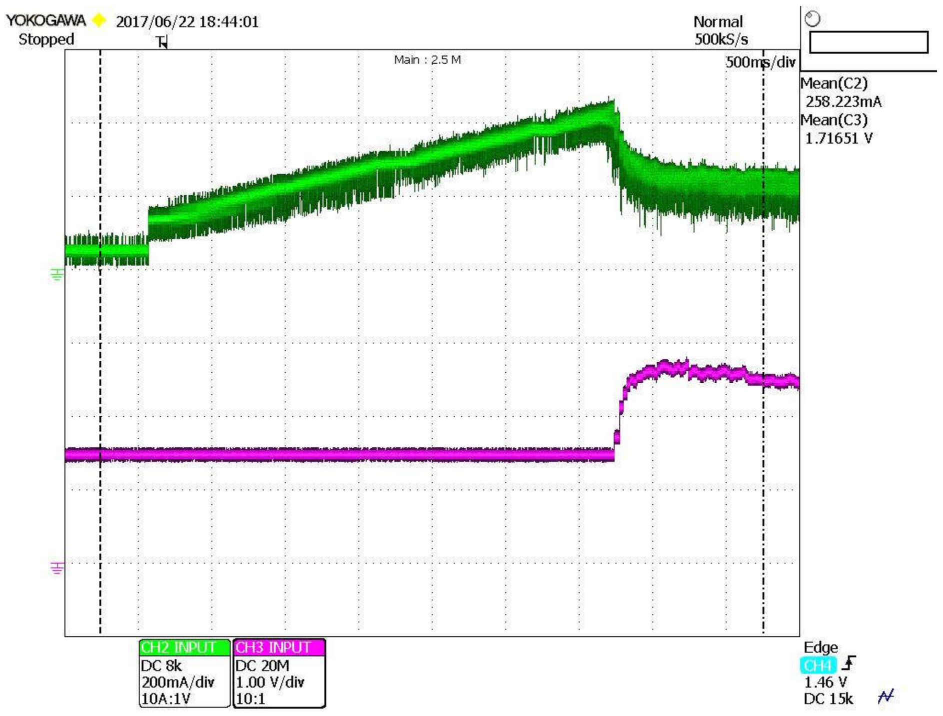


Fig. 5-11: Response to the motor start with the initial feed-forward relation. The green plot is the current and the pink one is the DAC proportional of the speed.

This graphic is really explicit and it perfectly shows the problem explained before. When the speed reference is received the controller begins to increase the current, but the motor speed stays null as the resistant torque is not passed. The current keeps growing until it achieves the 250mA required, when it falls back to the ideal value and the speed begins its response.

For avoiding this problem, the feed-forward relation is changed in order to correspond to the initial behavior of the motors. Doing this it is lost some effectiveness of the feed-forward once the motors are already moving, but it is not as critical as being completely inefficient when starting. Besides, this new relation remains closer to the real one so it still helps the control even when it is already started. The new feed-forward relation between the current and the speed is

$$I(w) = \begin{cases} 0.003w - 0.250, & w < 0 \\ 0.003w + 0.250, & w > 0 \end{cases} \tag{5.22}$$

With this new relation, the response of the control when exposed to the same conditions as before is much quicker, as showed on *Fig. 5-12.*

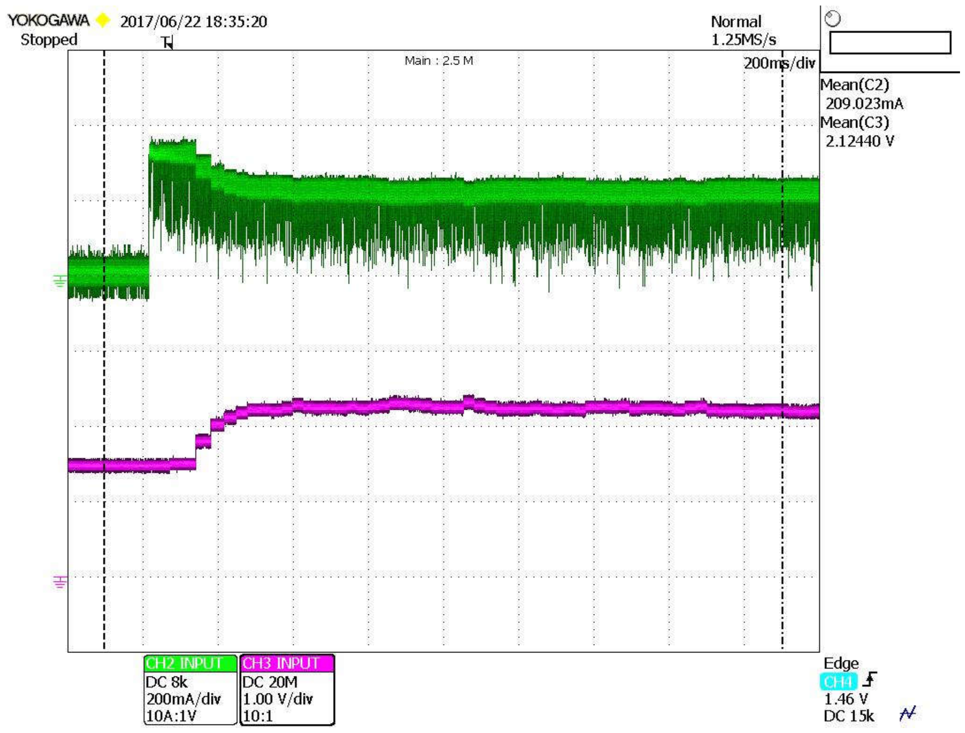


Fig. 5-12: Response to the motor start with the final feed-forward relation.

Now the current immediately jumps to 250 mA, overcoming the torque and starting the motor's movement. Note the huge time improvement: while before it needed more than 6s to achieve the desired speed value, now it does the same in no more than 200ms. This difference is not as big when the PI controller is quicker, but it is still noticeable enough to justify the addition of this feed-forward.

5.4.4. Experimental results

In order to analyze the correct behavior of the speed control it has been submitted to an exigent test that puts it to its limits. The reference speed has been configured using the microcontroller to periodically change from -5 rad/s to 5 rad/s. This way the control is exposed to high reference jumps and is constantly having to start the motors on each one of the directions, testing the most critical phase of the controlling process. This test is displayed on the oscilloscope on *Fig. 5-13.*

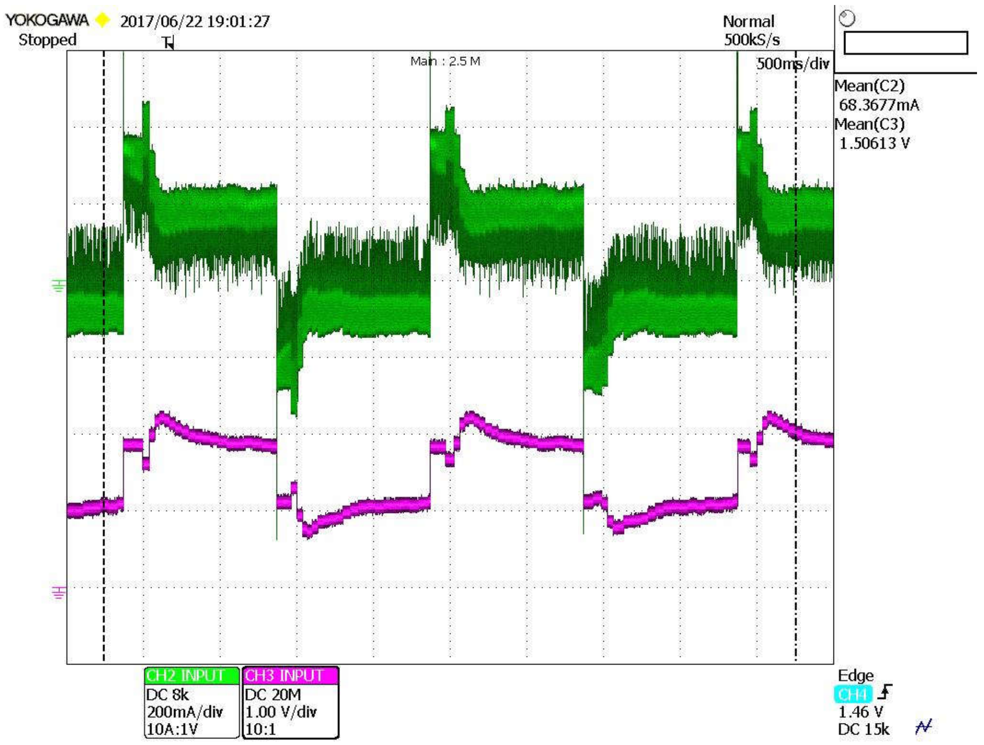


Fig. 5-13: Response of the speed control to a -5 rad/s to 5 rad/s reference. The green plot is the current and the pink one is the DAC proportional of the speed.

The control is working as expected, as both the current and the measured speed change their value and direction quickly. The current spikes correspond to the initial 250 mA needed to overcome the torque, as explained on the feed-forward section. The plotted speed signal is not continuous; this is caused by the fact that the encoders are not able to detect the direction of rotation so it is defined by the sign of the current and it immediately jumps together with it. However, the real speed signal is actually continuous, but it is not able to plot it without using the encoders.

The other experiment is made in order to test if the real response parameters correspond to the theoretical calculations made on the PI controller design. Choosing an establishment time of $t_s = 1s$ and an overweight of $S_p = 20\%$, the values of the PI constants are: $k_p = 0.023,\ k_i = 0.128$. With this parameters, the system response is captured on *Fig.5-14.* using an speed reference that goes from 10 rad/s to 5 rad/s and then back again to 10.

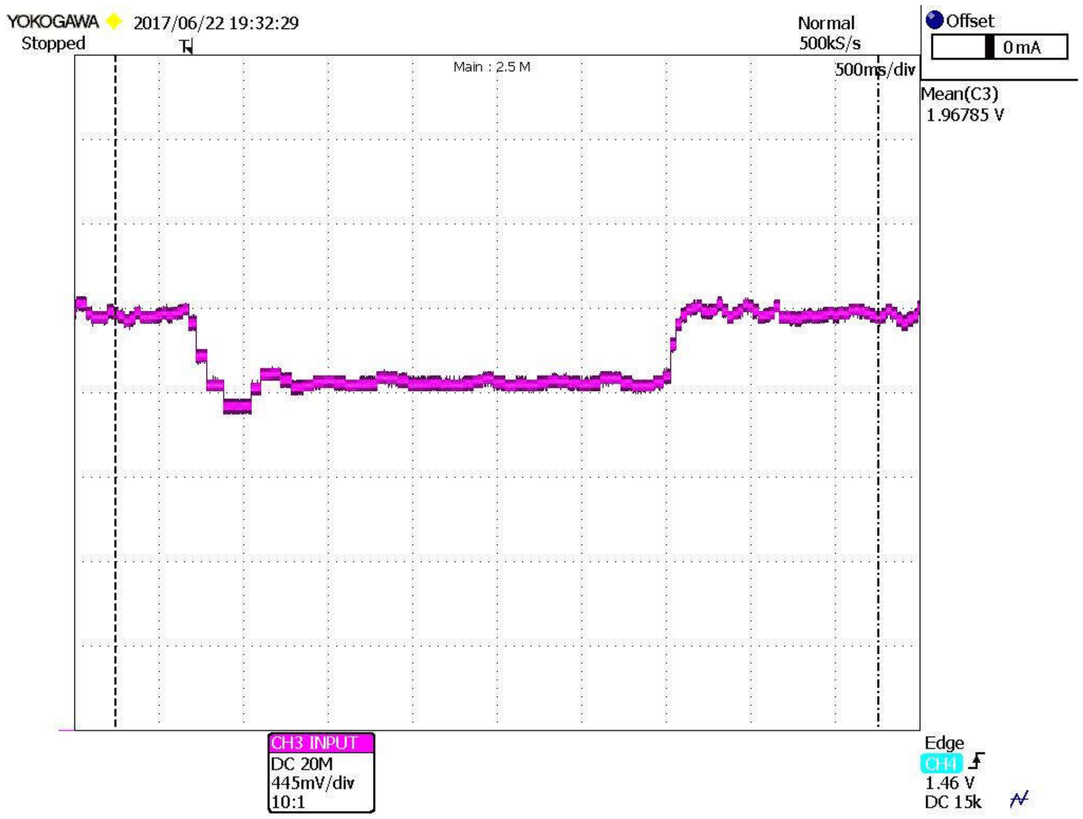


Fig. 5-14: Response of the speed control when configured to have $t_s = 1s$ and $S_p = 20\%$. Reference goes from 10 rad/s to 5 rad/s and back to 10.The plot is the DAC equivalent of the speed.

Analyzing the graphic, it is seen that when reducing the reference from 10 rad/s to 5 rad/s the response is really close to the theoretical calculations. The overweight of the first peak is approximately the 20% of the final gain and the time passed until the steady state is around two time divisions which equate to 1s.

However, the rising change from 5 rad/s to 10 rad/s doesn’t resemble the theory, as it has almost no overweight and the establishment time is really short. These differences can be attributed to the unpredictable behavior of the frictions of the motors and it must be assumed that not every response of the control will perform as calculated.

Conclusions

All the objectives settled at the beginning of the project have been achieved. A new two-wheeled vehicle robot has been assembled and it is managed using a current control.

The electronic circuits of the robot have been successfully understood, characterized and documented in the form of schematics. This task of documentation makes the construction of new robots much easier and removes the need for constantly checking the previous assembling. All the schematics are made using freeware so they can be easily modified on future projects.

Current sensors for the motors have been designed and implemented on the robot. They are able to capture the current intensity of the motors and transform it into a digital signal with notable precision. The sensors can be replicated and modified in order to obtain other currents of the robot, being useful for the implementation of other features.

The motors control have been designed, calculated and programmed using the current as the main control variable. This control is more precise than the previous voltage one, has a wider speed working range and a quicker response. The controllers are defined in a way that they can be easily changed to match the characteristics of the temporal response freely chosen.

Although at this point lots of projects have already been made on this matter, the two-wheeled vehicle robot line of investigation still has a bright future ahead. The next logical step would be to join this project with Carlos Conejo’s one, creating a system of current controlled vehicles communicating between them. Additionally, the use of current controls can be expanded to the trajectory control, unifying all the routines of the robot and reducing the number of controllers. As for the hardware improvements, using the schematics created on this project to design and build a Printed Circuit Board would be a really nice addition.

Acknowledgments

I would like to acknowledge my tutors Arnau Dòria-Cerezo and Víctor Repecho del Corral for all the help and experience given to me during the process of this demanding project. I can safely say that without Arnau's theoretical explanations and Victor's daily practical help at the laboratory I would not have been able to complete this challenge. In the same way, I want to say thanks to all the IOC team for making these five months a journey of learning and enjoying on a really healthy environment.

I also want to thank Iván Prats, Antoni Riera and Albert Costa for all the previous work that established the foundations that made my project possible. Furthermore, I also recognize the mutual help received by Carlos Conejo, working and learning hand by hand despite developing two really different projects.

Finally, I can't say farewell to this project without thanking my family and close people for all the support and encouragement given. The development of this project has had some up and downs, and the help of all the dears makes overcoming the hard moments much easier.

Bibliographic references

[1] Conejo Barceló,Carlos. *Adaptive Cruise Control, an scaled model: platooning using two-wheeled robots.* End of Degree Project UPC (2017).

[2] Prats Matinho, Ivan. *Control design and implementation for a line tracker vehicle*. End of Degree Project of Physics Engineering (2016).

[3] Riera Seguí, Antoni. *Disseny i implementació d'un sistema de comunicacions WiFi per a una xarxa de vehicles autònoms.* End of Degree Project UPC (2016).

[4] Costa Ruiz, Albert. *Design of controllers and its implementation for a line tracker vehicle*. End of Degree Project UPC (2017).

[5] [http://www.carminenoviello.com/2014/12/28/setting-gcceclipse-toolchain-stm32nucleo-part-1/]

[6] Yulin Zhang, Daehie Hong, Jae H. Chung, and Steven A. Velinsky, *Dynamic Model Based Robust Tracking Control of a Differentially Steered Wheeled Mobile Robot,* Proc. of the American Control Conference, (1998).

[7] P. Morin and C. Samson. *Motion control of wheeled mobile robots*. In B. Siciliano and O. Khatib, editors. Handbooks of Robotics, chapter 34, pages 779–826. Springer, (2008).

ANNEX: Code structure

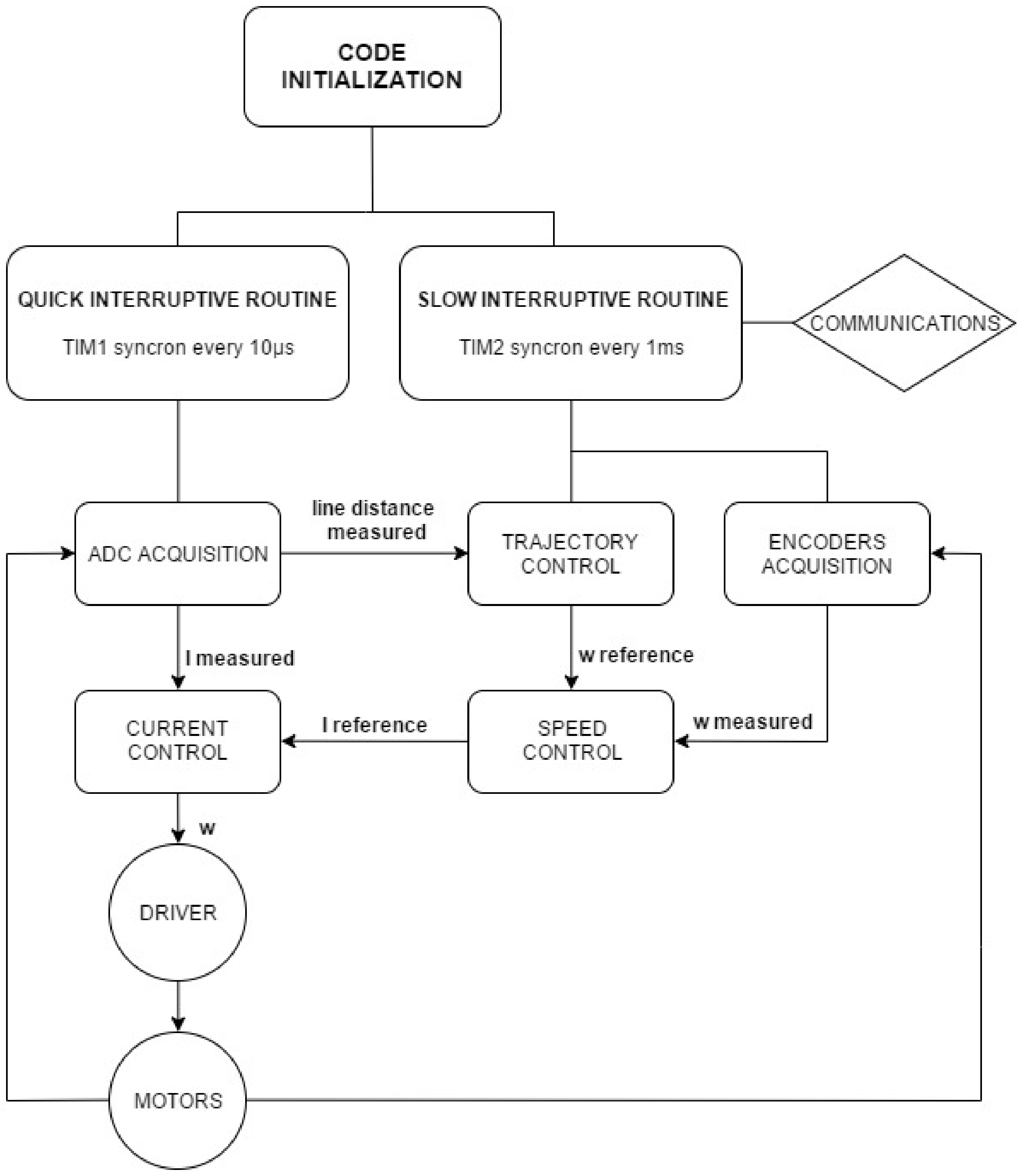


Fig. A-1: Block diagram of the code structure of the robot