Universitat de Vic - Universitat Central de Catalunya

Facultat de Ciències i Tecnologia (UST)

Double Degree in Industrial Electronics and
Automation & Mechatronics Engineering

Final Degree Project

A Programmable Six-axis Compliant Device based on a Gough-Stewart parallel platform

Sergi Martínez Sánchez

Director:
Dr. Federico Thomas

Director:
Sergi Hernandez

UVIC representative director:
Dr. Moisès Serra

September 2018

A Programmable Six-axis Compliant Device based on a Gough-Stewart parallel platform

by Sergi Martínez Sánchez

This project has been developed at Institut de Robòtica i Informàtica Industrial (IRI, CSIC-UPC) and presented at Universitat de Vic - Universitat Central de Catalunya in order to achieve the Industrial Electronics and Automation & Mechatronics Degree titles.

Director:
Dr. Federico Thomas

Director:
Sergi Hernandez

UVIC representative director:
Dr. Moisès Serra

A PROGRAMABLE SIX-AXIS COMPILANT DEVICE BASED ON A GOUGH-STEWART PARALLEL PLATFORM

Sergi Martínez Sánchez

Abstract

The goal of this project is to develop a system able to control the movement of a parallel platform according to the external forces applied to it. The project includes 1) the Gough-Steward parallel robot, created and assembled at IRI's lab; 2) 6 Dynamixel rotatory actuators, to move the platform; 3) an OpenCM 9.04 microcontroller board to get the reading of the forces provided by 4) a conditioning circuit for the 5) 6 load-cells located at each arms of the platform.

In this project it is described how the communication between the actuators, the electronic board and the computer, which acts as the master, is established. It is specified what kind, and how, the communication is performed between each device in order to send and receive data.

Finally, the proper kinematic study has been done to set the different configurations that the platform demands according the identification of the forces applied over the platform.

***

Acknowledgements

I would like to thank Federico Thomas for letting me join the IRI's team to develop this project, to Sergi Hernandez for the help and the tips he gave me, without him this project would not be possible. Thanks to his attention and trust, I have been able to finish a really complex and varied project overcoming the corresponding headages. To Mòises Serra for his constant concern and tracing of the project.
Finally, to my parents for their unconditional support and endless patience.

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

1.1 MOTIVATION

Nowadays, robots can be found in the fields of medicine, the military, the industry or even in the household sector. This leads to assume that robotics' field is expanding, involving in many other techno-scientific sectors and reaching, as well, the daily life. Drones, which were only used in the military industry, are proof of this. All this reflects the control we have achieved over robotics to this day.

Therefore, it is a field that expands the accessibility and capacity of the human beings in all possible and imaginable directions. Robotics, then, can be studied in all modalities, shaping them in different ways and creating robots of all sizes, appearances, and functionalities; making the analysis and the classification of robots a really diverse task.

This project, which leads the analysis of a 6 degrees of freedom parallel robot as a force stabilizer, will be my first little contribution to this huge community.

One of the main problems any engineer will have to face when it comes to develop a robot is the positioning one. For years many scientists have tried to solve it with different approaches, some like *odometry*[1] or *Kalman filters*[2] are the most used and successful ones, but even those introduce some accumulative error, that for long periods of time will increase until making the obtained position useless.

This project is my personal contribution to this problem, it aims to solve this problem by changing how the problem is focused. The idea behind the project is to instead of trying to reduce the accumulative error, accept that the error is already there. With this assumption I develop an adaptatively actuator which can work with some tolerances in the position error.

[1] Odometry is the use of data from motion sensors to estimate change in position over time.
[2] Kalman filter is an algorithm that uses a series of measurements observed over time and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone

1.2 STATE OF THE ART

This section shows the information that has been extracted from the available literature on the Gough-Steward platform. As far as the case is concerned, there are two great distinguished parts, the first objective is to control the platform to obtain any possible position. The second objective is for any configuration of the platform to be able to find the point of application of an external force from knowing the axial forces in the legs.

1.2.1 Gough-Steward Platform as an actuator

When it comes to controlling the platform to move it in any axis, either applying a translation or a rotation multiple different works have been found such a [10], interestingly all of them have in common the use of the kinematics to solve the problem.

Kinematics

Kinematics is a branch of classical mechanics that describes the motion of bodies without considering any force or torque involved. Usually, when applying kinematics to robots, they are modelled as systems formed by rigid bodies connected through articulations. The kinematics of robots describe position, orientation, speed, acceleration and all the derivatives of higher order than the position one.
Otherwise, dynamics treat the relations between the forces and torques presents in a system and the trajectories, speeds and accelerations of the elements that make it up.

In the robot we are studying in this project, the Gough-Steward platform, kinematics become the main component, leaving dynamics in the background. This is because, in the platform the acting force is very high in proportion to the inertial ones that can be produced. Considering such good characteristics and only needing to determine the desired orientations each time, the computing of the kinematics was the way to go. Furthermore, in a system of 6 DOF (degree of freedom) composed by 6 serial chains working in 3 dimensions, which each one will have 5 DOF and include 2 elements with singularities in its movement, will end up in an extremely complex system.

Therefore, considering kinematics the solution can be found following two possible directions:

- **Forward kinematics:** where we observe the response of the system to a given input.
- **Reverse kinematics:** where we obtain the motion plan according to the desired output.

Inverse kinematics Vs Forward kinematics

The problem of forward kinematics is in finding the final position of the actuator according to the translations and orientations of the actuators, whether, they are prismatic or rotating ones. For a serial chain i.e. is the product of the transformations from a reference point up to the end of the tool. This operation can be applied to describe any relationship between points if all the transformations that unite them are used, either symbolically or physically. For example, relationships for a transformation of reference 0 through 3 are of the type:

$$ {}^{0}T_3 = {}^{0}T_1 * {}^{1}T_2 * {}^{2}T_3 \quad (1) $$

For a series robot the solution of its direct kinematics is obviously unique since it does not contain any passive joints, all joints are actuated. On the other hand, for a parallel robot, the situation is much more complicated. A generic Gough-Stewart platform can have up to 40 solutions for its direct kinematics. By introducing alignments, symmetries, matches, etc., this number can be reduced to a minimum of 8 solutions for a 3-2-1 REF type manipulator [11].

The problem of inverse kinematics, on the other hand, consists on, from the orientation and position of the mobile platform, determine the values of the robot's actuators through the restrictions in each arm. In order to solve this problem, we deal with the localization of the ends of each arm to compute the inverse kinematics of each one.

It has already been showed that, in order to control the position of the platform, the values of the actuators that make it possible to reach a given position for the mobile platform have to be determined. The reverse kinematics of the designed robot must therefore be solved.

Representation of position and orientation

In order to solve the inverse kinematics of a parallel robot as the Gough-Steward platform its mandatory to understand first how each point of the platform is represented in the space.

The minimum number of coordinates to describe the position of a body in an Euclidean space is six. A reference ***i***, consisting of an origin expressed as O_i, and three orthogonal coordinate axes expressed as (x_i, y_i, z_i), are fixed in a given body. The position of this body will always be expressed relative to that of another, so it can be described as the position of a reference relative to another.

- Position of a body ***i*** relative to a reference ***j***:

$$ {}^{j}O_i = \begin{pmatrix} {}^{j}x_i \\ {}^{j}y_i \\ {}^{j}z_i \end{pmatrix} \quad (2) $$

- Orientation of a reference system ***i*** relative to ***j***, transforms a vector expressed in base ***i*** to base ***j***:

$$ {}^{j}R_i = (x_j \quad y_j \quad z_j) \cdot (x_i \quad y_i \quad z_i)^T \quad (3) $$

In the end, in order to represent a pose, a 4x4 where the fourth column represents the translation to be applied to the reference system ***i*** so that its origin coincides with the origin of the ***j***.

$$ {}^{j}T_i = \begin{bmatrix} {}^{j}R_i & {}^{j}O_i \\ 0_{1x3} & 1 \end{bmatrix} \quad (4) $$

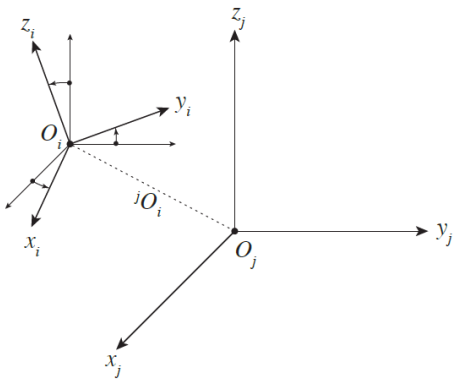


Figure 1 Reference frame i relative to j

1.2.2 Gough-Steward Platform as a sensor

The configuration of the Gough-Steward platform is not only reliable for allowing any possible movement, but it is also very interesting when using it as a sensor. Just by measuring the axial forces trough the legs it allows to measure different outer forces, very interesting works have been done before such in [1] or [2]. Nevertheless, in this project we are going to focus in the replicating the work done in [1].

Force/Torque sensor definition

Assume we have a platform in static equilibrium, connected to its environment through 6 legs, articulated with ball-and-socket joints Figure 1. Then, every leg applies a force f_i on the platform, which must be aligned with the leg and, by Poinsot's Central Axis Theorem, any other forces applied on the platform can be reduced to a single force F and torque acting along the same line l. We will next see how, having sensor readings of the leg forces, we can fully recover $\boldsymbol{F}$, $\boldsymbol{\Gamma}$ and the point $\boldsymbol{P}$ where l intersects the platform. In other words, this device can be used as a touch pad with force and torque feedback.

The sensor is formed by a platform connected to a base by means of six legs Each leg is joined to its ends by spherical joints and have installed a load cell through them in order to measure the longitudinal force that exercised. The spherical joints at the ends of the legs guarantee that the Force vector exercised for each one of them has the direction of the leg itself. In this way we have the six vectors strongly made by legs on the platform.

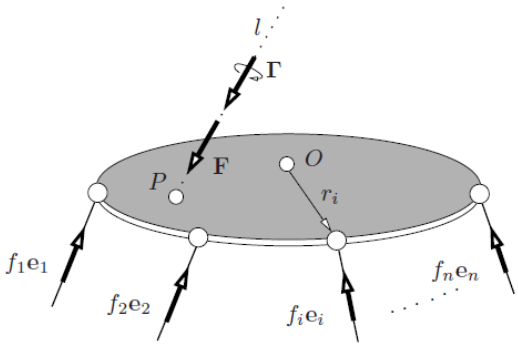


Figure 2 Representation of the vectors at the platform

Each leg applies a force f_i in the direction of the Euclidean vector of the leg $\vec{e}_i$. The resulting wrench $wl \in \mathbb{R}^6$, calculated relative to a reference point O solidarity to the platform's reference is:

$$wl = \begin{pmatrix} \sum_{i=1}^{6} f_i \, \vec{e}_i \\ \sum_{i=1}^{6} f_i \, \vec{e}_i \times \vec{r}_i \end{pmatrix} \tag{5}$$

Where $\vec{r}_i$ is the vector going from the reference point O to the ith leg attachment joint. We can write this in matrix form as $wl = J \cdot f$, where:

$$J_f = \begin{pmatrix} \vec{e}_6 & \dots & \vec{e}_6 \\ \vec{e}_1 \times \vec{r}_1 & \dots & \vec{e}_6 \times \vec{r}_6 \end{pmatrix}, \; f = \begin{pmatrix} f_1 \\ \vdots \\ f_6 \end{pmatrix} \tag{6}$$

Then the wrench can be expressed matricially,

$$wl = \begin{pmatrix} \vec{e}_1 & \dots & \vec{e}_6 \\ \vec{e}_1 \times \vec{r}_1 & \dots & \vec{e}_6 \times \vec{r}_6 \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_6 \end{pmatrix} \tag{7}$$

At this point we are going to suppose that in the moment of acquiring a measure the platform will be in static equilibrium, then, according to this supposition the sum of all external forces should be zero, therefore:

$$f_{plat} + wl = 0 \tag{8}$$

then:

$$f_{plat} = -wl \tag{9}$$

Where wl includes the forces performed by the legs and f_{plat} includes all the other forces carried out on the platform.

1.2.3 Contribution/Approach

In this project, therefore, it is chosen to give more complexity to the movement of the platform and give more versatility to its control. Thanks to the 6 degrees of freedom of the robot, the top platform can be placed anywhere in the space while at the same time the position of the force applied to it can be calculated. This will allow us to turn the platform into a compliant mechanism that is sensitive to external forces, which will be able to adapt its position based on the position and intensity of the force.

To be able to supply all the objectives set, the construction of a system consisting of multiple parts is carried out. This will be a complex system formed by different hardware running a specific firmware, which are communicated within the same bus with different layers of software running within a PC.

1.3 GOAL

The goal of this project is to build a system able to control the configuration of a Gough-Stewart parallel platform robot using the forces applied to it. The platform must remain in its horizontal configuration when no force is applied, which is parallel to the floor reference. If any external force is applied, then, the platform must tilt or move a distance proportional to the applied force. To achieve this goal, we will work on:

1. Six load cells placed on each platform's arms, to measure the axial forces in order to find the application point of the force.
2. An electronics board that combines the analogic conditioning circuit for the load cells and the corresponding microcontroller to get the data.
3. The proper low-level programming code to run the microcontroller.
4. Understand how the Dynamixel communication protocol works to apply it.
5. Develop the corresponding high-level drivers in order to establish a communication with the boards.
6. A platform controlled by a 6-RUS manipulator which will set the parallel platform position.
7. The computer application responsible of processing all the data and send the corresponding instruction.

Once the goal of the project is defined and the state-of-the-art is revised, it is possible to think about which elements will be needed to develop the project

1.3.1 Block diagram

Figure 3 shows how the main parts are combined to form the entire system, the Gough-Steward platform interacts with the conditioning board as they read the axial forces, and with the actuators as they act through a mechanical movement. Finally, both of them stablishes communication with the PC using an asynchronous serial bus.

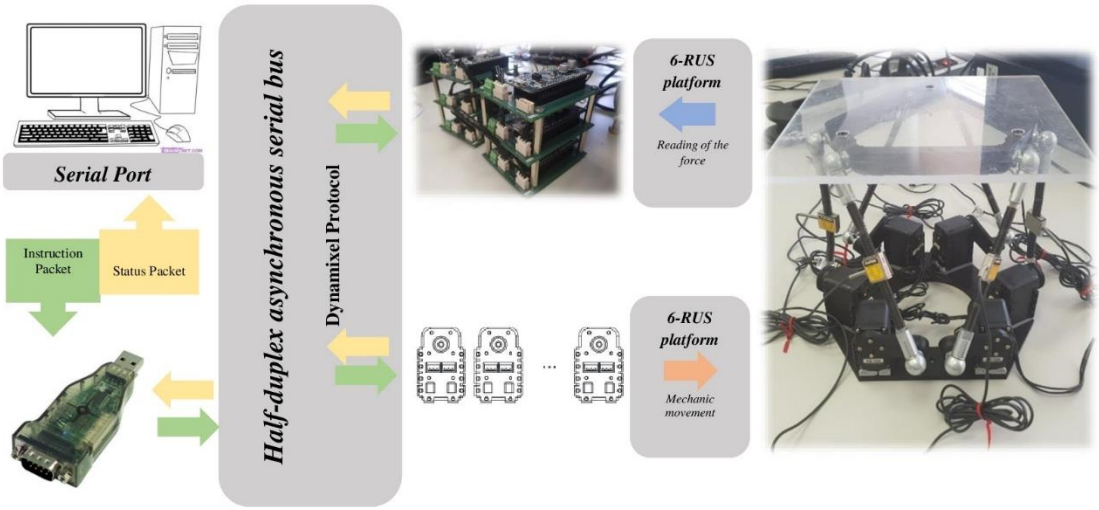


Figure 3 Block diagram

2 TOOLS

Once the objectives of the project are clear, the definition of the tools used in the project was needed. The used elements are essentially distinguished in two categories: the hardware and the software.

The chosen tools used in the project where highly conditionate by the advice given by the technicians at IRI, their contact with suppliers and their personal experience made decisive the election of the devices and the programs that were finally used in this project.

Nevertheless, different alternative elements and methods can be used to obtain the same result.

2.1 HARDWARE

This section explains the different elements that are part of the hardware, those are: the main platform which makes the body of the device; 6 actuators, which execute the orders to move the platform; a microcontroller board to get the data provided by the load cells; a printed circuit board that includes the circuit for conditioning the signal and an USB interface to communicate the microcontroller board and the actuators with the CPU.

2.1.1 The Gough-Stewart parallel robot

The main aim of this project is to sense the application of a force and control the orientation of a platform. In order to accomplish the purposed goal, the body of the device had to allow us to know the point of application of the force and move it proportionally in any direction, therefore, the Gough-Stewart parallel platform configuration was chosen for two reasons:

1. The Gough-Stewart parallel offers 6 DOF (3 translations and 3 rotations) which means that it allows any movement in the space. This possibility is mandatory to develop the desired actuator as the forces can be applied in any point of the surface.

2. As it has 6 legs, when measuring the axial forces trough the arms, the Gough-Stewart parallel platform offers the minimum configuration to compensate the arbitrary position of the force and torque applied to its surface, allowing us to find the original point of application.

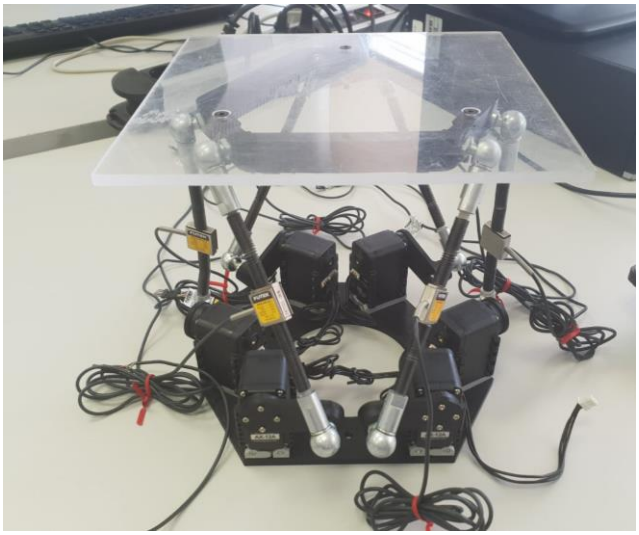


Figure 4: Device body

From (Figure 4: Device body) we can see that the platform used does not exactly correspond to a Gough-Stewart parallel platform, as the original one uses sliding bars as actuators and this one uses rotatory actuators. This variance is called 6-RUS and introduces a slight change in the geometry that must be considered when it comes to the kinematics calculations.

Definitions

The Gough-Stewart platform is a six-degree-of- freedom parallel manipulator. It was first proposed for the testing of wear in tyres by Gough and later by Stewart as a flight simulator. The Gough-Stewart platform manipulator consists of a moving platform connected to a fixed base by means of six extendable legs. Controlled extension and/or retraction of the legs results in six-degree-of-freedom motion of the moving platform. This feature provides the platform which higher versatility than other parallel robots, as it is able to perform a rotation and translation in all three axis of the space.

Advantages

The main advantages of a Gough-Stewart platform manipulator are its increased load carrying capability since the load acting on the moving platform is `shared' by the six legs.

As a parallel manipulator the Gough-Stewart platform is design with the combination of kinematics chains of few simple articulations, thus, the platform shows a higher rigidity when receiving an external force than a serial manipulator.

The positioning errors in the kinematic chain get highly reduced due to the whole average of the other kinematic chains, on the other hand serial manipulators show an accumulative error in every kinematic chain.

Furthermore, the moving platform is very light in comparison with the base, where the actuators are placed, then the resulting wrench caused by the inertia of any movement is very low and usually makes despicable the dynamics of the parallel robot.

All those facts result in a robot much more rigid, precise, capable of carrying more weight and reaching higher accelerations and speeds than serial manipulators.

Figure 5 Some Gough-Stewart platforms from IRI's lab

Architecture

The parallel robot prototype developed at IRI's lab has some relevant characteristics, since it is designed to be a portable device. First of all, the robot requires to have little dimensions to be portable. As reminded in the Advantages of parallel robots, this fact largely reduces the complexity of the problem, as the dynamics of the robot are not considered. On the other hand, for this application the platform needs run as fast as possible to react in real time to the forces, which means that the actuators should operate at a significantly high speed. As the consequence, the manipulator should have simple kinematic chains to be easily controlled.

The original architecture of the Gough-Stewart comes with the use of prismatic actuators. This is a very common type of actuator when it comes to parallel manipulator as it eases the kinematics calculations and offers rigidity to the structure. In this case the kind of parallel manipulator used is called 6-RUS, and the serial arms attached to the actuators dispose of two spherical joints, one at each end, which provides an extra degree of freedom. Therefore, the arm can move in a three-dimensional space to guarantee the 6 degrees of freedom of the platform. In addition, the initial setup of the actuators' rotors provides faster movements to the platform in their home position, which means that reaction times are smaller than in other manipulation systems.

As a counterpart, the serial kinematic chain of this kind of manipulator introduces serial singularities. They arise when the actuators' rotor and the arm acting as connecting rod get aligned. In this disposition, the mechanism has two feasible ways to carry on the movement and the system suffers from a loss of performance.

2.1.2 Actuators

In order to move the platform 6 actuators were needed, the chosen ones are the Dynamixel AX-12+ (Figure 6 left), from the AX series of the manufacturer ROBOTIS. Those actuators are high-performance and versatile servomotors ideal for small robotics applications, which can be found in a lot of low cost humanoid robots. The fact that they are servomotors distinguish them from DC motors as they allow to get the position where they are, and command them to move a certain angle (absolute or relative to its position) thanks to a rotary encoder and an embedded controller (Figure 6 right).

The most relevant characteristics are:

- Resolution: 0.29º
- Potentiometer: 10 bits, 0º - 300º range
- Motor: Cored
- Communication Speed: from 7343 bps up to 1 Mbps
- Feedback: position, speed, temperature, load, input voltage and current.
- Communication protocol type: half-duplex Asynchronous Serial Communication (8 bits, 1 stop, no parity)

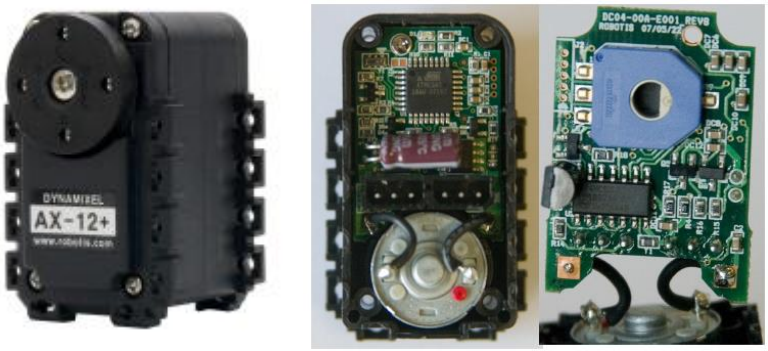


Figure 6: The Dynamixel™ AX-12+ rotary actuator & its electrical components

The rotatory encoder is an electromagnetic device used to get the angular position of the axis. On the other hand, the embedded controller manages the control system of the motor and updates the information in the RAM and EEPROM registers where the data of the actual status of the actuator is stored.

The EEPROM register is used to store data that must remain saved although the device is turned off e.g. the ID, the Baud Rate of the communication or the return time between packets. Otherwise, the RAM register is used to show the actual status of the actuator such as the current position, the current speed or the temperature, the RAM register is also used to set the orders e.g. goal position, goal speed or torque limit.

By accessing those registers is possible to govern the servos as desired, to do it, either read or write data on the registers, a communication protocol established by the manufacturer must be used, this protocol is called Dynamixel™ communication 1.0.[3] The remaining use of this protocol of communication will be deeply discussed in Chapter 2.2.1.

The Dynamixel™ AX-12+ has two operation modes: the joint mode (thought to set a position of destination between 0º and 300º), and the wheel mode (to allow endless turn at a specified speed).

The servomotor will be used in joint mode, which means that the actuators will request the angular positions to move to. The possible Goal Position values that the actuator can get in joint mode are compressed in a range from 0 to 1023. The angular range and the configured angular position settings for each actuator are shown in (Figure 7).

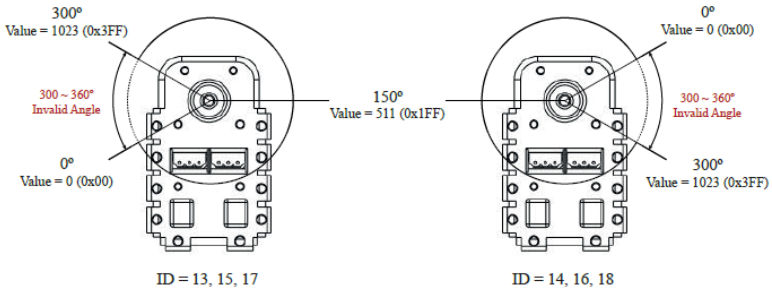


Figure 7: Possible configurations

[3] Dynamixel™ protocol communication is specified in here
http://support.robotis.com/en/product/actuator/dynamixel/communication/dxl_packet.htm

2.1.3 Microcontroller Board

In order to transmit the readings from the load cells to the computer, a microcontroller was needed, the microcontroller should have at least a 12 bits ADC (3.3v Ref.) to read the signal with the desired resolution and I^2C communication capabilities to send instructions to the peripherals in the board, these parameters match with most of the microcontrollers available in the market.

Therefore, the OpenCM9.04 microcontroller board (Figure 8) was chosen.

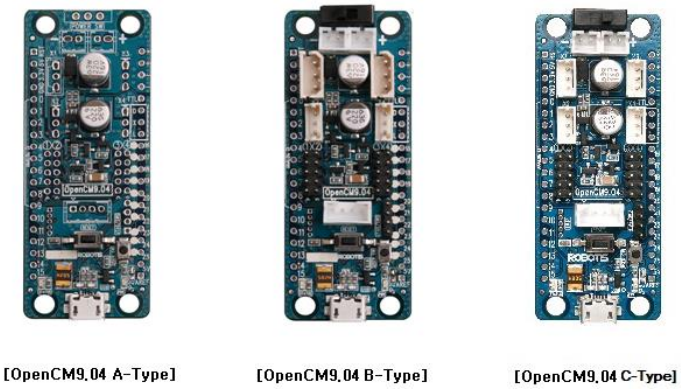


Figure 8 OpenCM 9.04 board types

The OpenCM9.04 is an open source. Low cost, microcontroller board based on 32bit ARM Cortex-M3 developed by ROBOTIS the same manufacturer as the actuators. The choice of using this board was based (apart of accomplishing the previous requirements) taking into consideration the following aspects. Firstly, the fact that as this board is from the same manufacturer as the actuators makes it ready to stablish a communication with a Dynamixel bus. Therefore, it includes the peripherals required to use the half-duplex serial communication protocol, this would allow us to use the same communication bus to either set the commands to the actuators and read the values of the load cells, reducing the number of ports used in the computer to one. The second one was the fact that the manufacturer offers Arduino libraries allowing us to program it through the Arduino IDE simplifying the difficulty. Finally, the previous experience from the technicians at the IRI encourage this decision.

The main functions of the OpenCM9.04 microcontroller board will be:

- Get the conditioned signal from the load cells at predefined sample frequency.
- Control the board peripherals to compensate the amplification offset
- Act as a slave device in the half duplex communication bus to send the required data when asked.

2.1.4 USB interface

As explained before the computer is the responsible of getting all the information provided by the microcontroller trough the Half-Duplex communication bus, process the data and send the corresponding instructions to the actuators trough the same bus.

To be able to communicate the computer with the Half-Duplex serial communication bus an adapter was needed. The USB2Dynamixel (Figure 9) from the manufacturer ROBOTIS was chosen. This adapter offers the serial conversion from USB or RS232 (serial port) to TTL (Half-Duplex) / RS485 (serial communication).

Figure 9 USB2Dynamixel Adapter

A serial communication is a standard communication protocol that uses a data transmission channel where the data is send transmitting each bit individually. This adapter is used to transform the data received by the USB to the TTL port (required for the Half-Duplex communication) and vice versa. The TTL port consists of 3 pins: power, ground and data, this allows the communication to happen in only one wire and connect more devices in the same bus.

2.1.5 Evaluation Board

The evaluation board MCP6N16 from the manufacturer microchip has been used in order to stablish a first touch with the instrumentation amplifier MCP6N16, this board comes with differential input filtering, two jumper selectable gain settings and output filtering, in addition to an external voltage reference circuit to allow for an adjustable output common-mode level shifting.

This board has been used in order to test the performance of the instrumentation amplifier and verify the proper calculus done in Chapter 3.

2.2 COMMUNICATION PROTOCOLS

This section explains the different communication protocols used in the project, those are: the Inter-Integrated Circuit used to control all the peripherals in the circuit, and the Dynamixel communication protocol used to communicate the actuators and the conditioning boards .

2.2.1 Dynamixel

In order to stablish the communication between the computer and the platform the need for a communication protocol was obvious, the most common solution for this kind of applications is to use the serial communication, but as in this project we are using Dynamixel actuators as explained in Chapter 2.1.2 the use of the protocol imposed by the manufacture was unavoidable, thus, we decided to use the same protocol to implement the communication with the conditioning boards.

This protocol works by following the hierarchy of master-slave, the master, in our case the PC, sends an instruction packet to any slave device, in our case either an actuator or a Conditioning Board, those will read the packet, then execute the corresponding instruction and finally will return a status packet with the information corresponding to the instruction. It's important to highlight that in order to specify the target of the packets, every slave will have an identifier which will be part of the packet, therefore, only when a slave device reads an instruction packet with its identifier will execute the instruction.

Even though this protocol was developed by ROBOTIS, the communication itself remains as a binary communication based on Serial Communication using hexadecimal base.

Instruction packets

As explained before the instruction packet will be the packet sent by the PC to any slave device and it will contain an order to execute.
The general structure of an instruction packet is:

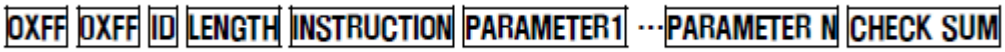


Figure 10General instruction packet

Each section represents a byte [4]of information in hexadecimal code, the meaning of each byte stands as follows:

- **0xFF 0xFF:** This signal notifies the beginning of the packet
- **ID:** It is the identifier of the Dynamixel device which will receive Instruction Packet
- **Length:** It is the length of the packet. The length is calculated as "the number of Parameters (N) + 2".

[4] The byte is a unit of digital information that most commonly consists of eight bits

- **Instruction:** This command gives an instruction to the Dynamixel device and has the following types:

Value	Name	Function	No.of Parameters
0x01	PING	No execution. It is used when controller is ready to receive Status Packet	0
0x02	READ_DATA	This command reads data from Dynamixel	2
0x03	WRITE_DATA	This command writes data to Dynamixel	2 or more
0x04	REG WRITE	It is similar to WRITE_DATA, but it remains in the standby state without being executed until the ACTION command arrives.	2 or more
0x05	ACTION	This command initiates motions registered with REG WRITE	0
0x06	RESET	This command restores the state of Dynamixel to the factory default setting.	0
0x83	SYNC WRITE	This command is used to control several Dynamixels simultaneously at a time.	4 or more

Table 1 Types of Dynamixel Instructions

- **Parameter 0…N:** Parameter is used when Instruction requires ancillary data.
- **Check Sum:** It is used to check if packet is damaged during communication. Check Sum is calculated according to the following formula.

$$Check\ Sum = \sim(ID + Length + Parameter1 + \cdots Parameter\ N)$$

Where, "~" is the Not[5] Boolean operator.

Status packets

As explained before the status packet will be the packet sent by any slave device as an answer to an instruction packet.

The general structure of an instruction packet is:

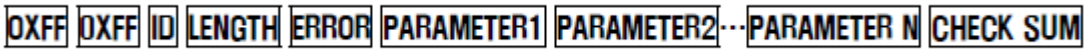


Figure 11 General Instruction packet

The status packet shares the same structure as the instruction packet except for the error byte and the parameter ones.

- **Parameter 0…N:** This byte returns data only when READ_DATA instruction is send, in case of any error it will be empty.
- **Error:** It displays the error status occurred during the operation of the communication.

[5] Not operator: In Boolean algebra, the NOT operator is a Boolean operator that returns TRUE or 1 when the operand is FALSE or 0, and returns FALSE or 0 when the operand is TRUE or 1

Bit	Name	Contents
Bit 7	0	-
Bit 6	Instruction Error	In case of sending an undefined instruction or delivering the action command without the reg_write command, it is set as 1.
Bit 5	Overload Error	When the current load cannot be controlled by the set Torque, it is set as 1.
Bit 4	Checksum Error	When the Checksum of the transmitted Instruction Packet is incorrect, it is set as 1.
Bit 3	Range Error	When a command is out of the range for use, it is set as 1.
Bit 2	Overheating Error	When internal temperature of Dynamixel is out of the range of operating temperature set in the Control table, it is set as 1.
Bit 1	Angle Limit Error	When Goal Position is written out of the range from CW Angle Limit to CCW Angle Limit , it is set as 1.
Bit 0	Input Voltage Error	When the applied voltage is out of the range of operating voltage set in the Control table, it is as 1.

Table 2 Dynamixel Error bits

2.2.2 I^2C

As explained in Chapter 2.1 the Conditioning Board is a board that combines both the conditioning circuitry with the necessary peripherals and a microcontroller board. The OpenCM 9.04 board provides different kinds of communication protocols: the Serial Peripheral Interface protocol or SPI, the Serial Communication through the serial port and the Inter-Integrated Circuit protocol or I2C. Then the I^2C interface was chosen as it is the one that allows the communication to work with less connections, rejecting the need of using SSEL/CS lines.

The I2C interface is a two open-drain and bi-directional connection consisting of the following signals: serial data (SDA) and serial clock (SCL). In generalized I2C implementations, attached devices can develop two basic roles, i.e. the master, device which puts an address in the bus in order to interact with the slave device with the matching address, which acknowledges the master.

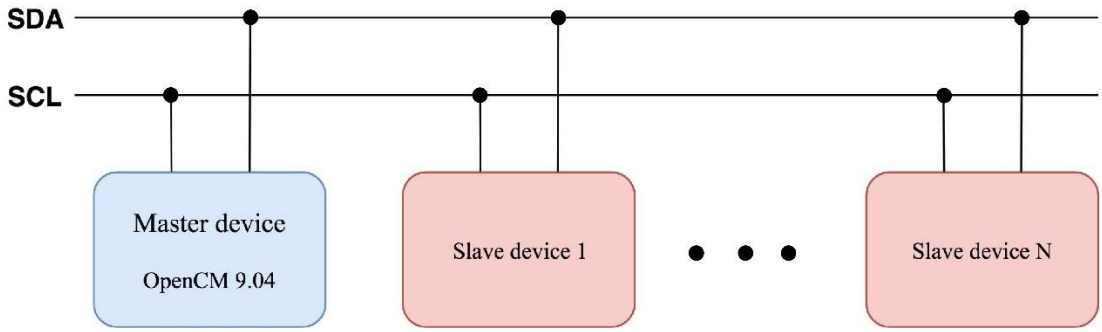


Figure 12 I2C protocol diagram

The involved communication bits develop their own function depending on which line (SDA or SCL) they are transmitted.

- **SDA:** This line is where all the exchanged words between master and slave are transferred.

- **SCL:** This line broadcasts a clock signal which synchronizes the word exchange on the bus. This signal is generated by the master device.

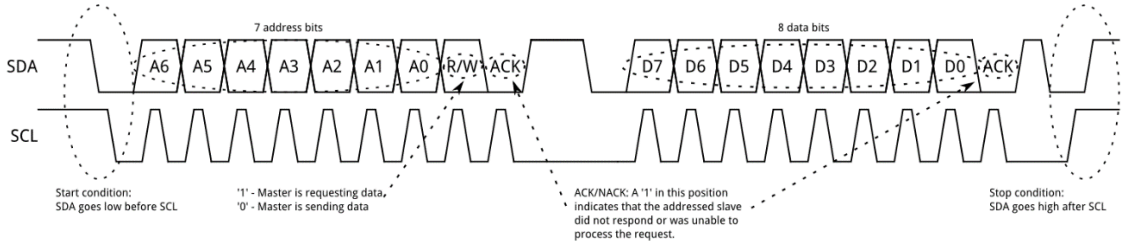


Figure 13 I2C protocol

As it has been said before, these two lines are open-drain, i.e. pull-up resistors need to be attached so the lines are set to HIGH. That fact is important to understand how data signals are transferred. The following figure shows how a conversation between master and device looks like.

Start Condition

To initiate the broadcast, a starting condition needs to befall: The master initiates data transfer by establishing a start condition when a high-to-low transition on the SDA line occurs while SCL is high.

Address Frame

The address frame is always first in any new communication sequence. For a 7-bit address, the address is clocked out most significant bit (MSB) first, followed by a R/W bit indicating whether this is a read (1) or write (0) operation.

The 9th bit of the frame is the NACK/ACK bit. Once the first 8 bits of the frame are sent, the receiving device is given control over SDA. If the receiving device either did not receive the data correctly or did not know how to parse it, the NACK/ACK bit is set to high.

Data Frames

After the address frame has been sent, data can begin being transmitted. The master will simply continue generating clock pulses at a regular interval, and the data will be placed on SDA by either the master or the slave, depending on whether the R/W bit indicated a read or write operation.

Stop condition

Once all the data frames have been sent, the master will generate a stop condition. the master device must leave the SCL signal HIGH while the sending device pulls SDA signal from 0 to 1.

2.3 SOFTWARE

This section explains the different tools used in this project that are part of the software, those are: OrCAD the program used to simulate the circuit design and design the PCB; the Arduino IDE used to program the microcontroller board; Matlab and its Robotics toolbox used to verify the kinematics calculations and the libraries provided by the IRI to communicate with the devices with the Dynamixel protocol.

2.3.1 OrCAD

OrCAD is a software tool suite used to create electronic schematics and electronic prints for manufacturing printed circuit boards. OrCAD is a suit composed of different products:

- OrCAD Capture: is the tool used to come up with the electronics design of the circuit. Capture does not contain in-built simulation features, but exports netlist data to the simulator, OrCAD EE.
- OrCAD EE PSpice: is a tool that incorporates SPICE circuit simulator allowing the suite to simulate the circuits.
- OrCAD PCB Designer: is a printed circuit board designer application.

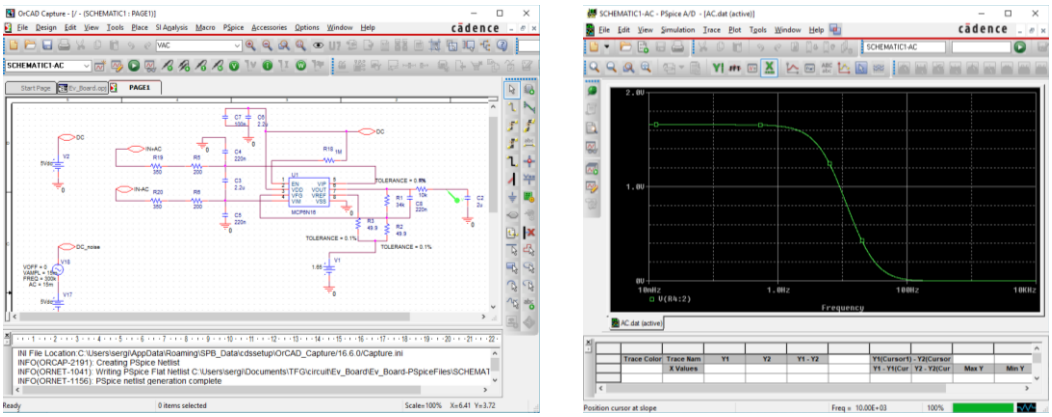


Figure 14 OrCAD Capture + EE PSpice

2.3.2 Arduino IDE + ROBOTIS Dynamixel libraries

The software of the Arduino® electronic boards provides an Integrated Development Environment (IDE) based on the Processing language. This software is used to program the OpenCM 904 board, although the Open CM904 board is not an Arduino Board it comes preprogramed with a bootloader that allows to program it from the Arduino IDE, to do it the manufacturer provides a series of libraries[6] that can be installed in the Arduino IDE through the Board Manager.

[6] Arduino OpenCM 9.04 ROBOTIS libraries: http://emanual.robotis.com/docs/en/software/arduino_ide/

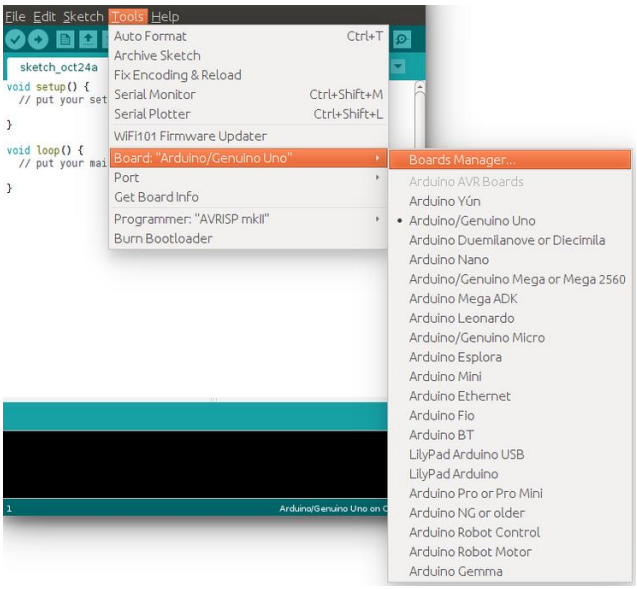


Figure 15 Arduino IDE (Board Manager)

Even though the libraries provided from the manufacturer are very extended and offers a lot of functionalities, all of them are oriented to use the board as master of the Dynamixel bus, and, as we want to use the Board as slave we will have to develop our own library to make it behave as a slave. This process will be deeply discussed in chapter 4.1 Firmware.

2.3.3 Matlab Robotic toolbox

Matlab® software is a powerful mathematic tool, with its own programming language, that has been absolutely helpful in lots of different aspects involving this project.

All the mathematical work done, and explained in Chapter 4, and all the simulators have been developed using this software as the implementation and verification of calculations becomes much faster and easier. Among the basic benefits of the Matlab® software for the development of this project are:

- Matrix manipulation
- Data representation through lineal and superficial graphics using 2D and 3D plots
- Possibility of simulations

Besides, toolboxes can be added, as the Robotics Toolbox[7] , developed by PeterCorke and used in section 4.1. Once the Robotics toolbox is downloaded and added to the Matlab®'s path, it is ready to be used. This toolbox offers algebraic help for all the matrices transformations needed to set the kinematics in a really simple and understandable way.

The library offers the functions to easily perform the 3 rotations and 3 translations by using the functions (trotx(), troty(), trotz() and transl()). As well, the library offers some facilities to plot the corresponding points in the 3D space.

[7] Matlab Robotics toolbox: http://petercorke.com/wordpress/toolboxes/robotics-toolbox

2.3.4 Dynamixel libraries from IRI

In order to use the USB2Dynamixel adapter and communicate to the devices in the Dynamixel bus with a code running in a PC some libraries are needed. Even though the manufacturer ROBOTIS also provide their own libraries[8], in this project we are going to use the libraries developed by the technicians at the IRI, those libraries are Open Source libraries developed in C++ for any Linux OS. Among all the libraries developed at IRI we are going to use the following ones:

- **Iriutils:** This set of tools provide basic functionalities that are generally needed in many applications. These utilities include mutual exclusion objects, exceptions, threads logic events and also generic communication devices.
- **Comm:** This is a set of drivers for standard communication devices. This library offers a generic communication device which can be used to create drivers for new communication devices easily: it is only necessary to implement the low level, operating system dependent functions, all the other features are already implemented.
- **Dynamixel:** This library provides a server to handle all the devices connected to a Dynamixel Bus and also low level read and write functions to access each one of these devices.

  The server allows to:

  · Scan the bus for devices: it checks all the possible addresses (from 0 to 254) for all the possible frequencies. Once a device responds in a given frequency, all other frequencies are discarded.

  · Change the ID of a given device on the bus. By default, all devices came with the same address (0) and it is necessary to change it in order to have multiple devices on the same bus.

  · Change the bus baudrate. Some devices have a slow baudrate by default and to achieve the peak performance it is necessary to change it.

  For each device, the following operations are provided:

  · Read and write to any 8 or 16bit register. Allow access to all the device registers.

  · Synchronized write operations. Allow loading a given value to a given register to all the devices at the same time. The value is sent to the device using a REG_WRITE operation, but it does not take effect until the ACTION command is sent.

  · Broadcast a command to several devices.
- **Dynamixel Motor Controller:** this library uses the previous one (Dynamixel library) to control all the parameters of a Dynamixel actuator, it eases the task of setting a goal position, configure the actuators and tuning the PIDs. It also allows the possibility of creating a group of motors in which a different goal position can be set at the same time, this capability will be really useful in our application.

[8] Dynamixel C++ libraries: http://support.robotis.com/en/software/software_development_kit.htm

Cmake

In order to compile the libraries an all the code we are going to develop the tool CMake is needed.

CMake is a script language used to automate the process of creating Makefile files for a program. It uses a file on each folder (CMakeList.txt) which tells the Make application what to do with the files in that level, and can also call scripts from any sub-folders to handle the files in them.To start the process, CMake executes the CMakeList.txt file in the root folder of the program, which in turn, executes the CMakeList.txt files in any of the sub-folders.

The result of this process is a Makefile file which can be used to generate the desired binary or library.

2.3.5 Eigen

One of the main characteristics of this project is amount of matrix calculation needed either for computing the inverse kinematics, the force/torque sensor or the Kalman filter. Therefore, the code developed in C++ will require to realize a lot of calculations using matrix which can result in a very tedious task. In order to ease the task of computing these matrices we decided to look for a tool that manages all the matrix computation.

Eigen is a high-level C++ open source library of template headers for linear algebra, matrix and vector operations, geometrical transformations, numerical solvers and related algorithms.

It has been chose because of its versatility (it supports all matrix sizes, all standard numeric types, including std::complex), reliability, elegance and speed.

2.3.6 QT

In order to create the final graphic user interface a framework capable of running in Linux, designed to build GUIs and that allows us to add all the C++ code developed previously was needed, this is QT.

QT is a cross-platform application framework and widget toolkit written in C++ for creating classic and embedded graphical user interfaces, and applications that run on various software and hardware platforms with little or no change in the underlying codebase, while still being a native application with native capabilities and speed.

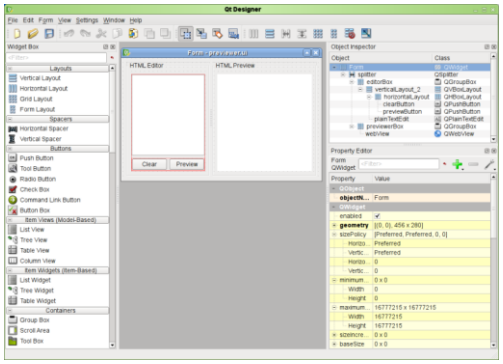


Figure 16 QT IDE

Qt comes with its own Integrated Development Environment (IDE), named Qt Creator.

3 ELECTRONIC DESIGN

This chapter addresses the problem of designing and building a circuit that combines both analogic and digital electronics. The solution implemented in this project to acquire the information provided by the six load cells placed in each arm is to build six identical boards which contains the conditioning circuitry and a microcontroller to provide the data to the PC.

Any circuitry design starts by realizing to main process, first the circuit design and then the design of the printed circuit board. Both of them are really important in order to achieve a good design and a hardware that provides the desired performance, as, even a small mistake at any point can cause big errors in the final result.

3.1 BLOCK DIAGRAM

The flow work of this circuit can be explained through the following block diagram.

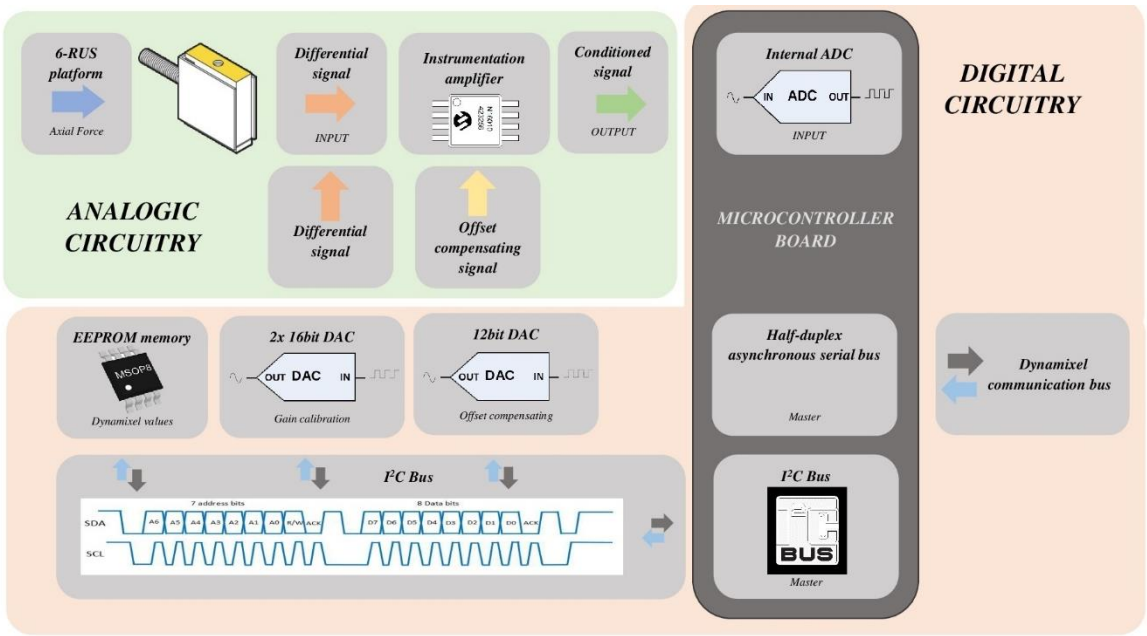


Figure 17 Circuit Block Diagram

First of all, it is mandatory to clarify that the circuit has two main parts that need to be combined, one is the analogic circuitry, which is in charge of managing the analogic signal provided by the Load Cell. The other one is the digital circuitry which involves the microcontroller board, the digital communication to the Dynamixel bus and the external peripherals used.

From Figure 17 we can see that when some force is applied to the Load cell it generates a differential signal that is fed to the instrumentation amplifier in order to conditionate de signal. Once the signal is conditioned it will be read by the internal ADC of the microcontroller board. At the same time the circuitry involved around the instrumentation amplifier needs a variable signal to compensate the offset, this signal is provided by a 12bit DAC controlled by the microcontroller through the I^2C bus, in the same bus there are two 16bit DACs used to artificially generate a differential signal to calibrate the gain error of the circuit, as well as an EEPROM memory used to store the values needed to emulate a Dynamixel slave. Finally, the microcontroller board uses the half duplex buffer hardware to establish the asynchronous serial communication with the Dynamixel protocol.

3.2 COMPONENTS

Once the work flow of the circuit is stablished the choice of the components to use was mandatory, the following sections will explain the characteristics of the main components that where choose.

3.2.1 Load cell

As the main objective of this circuit is to measure the axial force present through the arms of the Gough-Stewart platform, a transducer capable of converting force into an electrical signal was necessary, thus, the most common transducer to perform this task is the load cell.

A load cell is a transducer based on a mechanical element of which the force is being sensed by the deformation of a (or several) strain gauge(s) on the element.

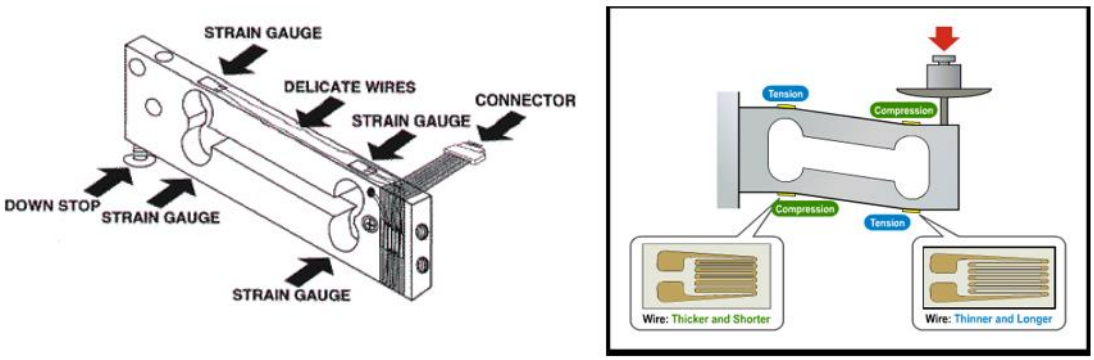


Figure 18 Load cell example

A strain gauge is a device that measures electrical resistance changes in response to, and proportional of, strain applied to the device. The most common strain gauge is made up of very fine wire, or foil, set up in a grid pattern in such a way that there is a linear

change in electrical resistance when strain is applied in one specific direction. Then by placing the strain gauges in a material with a known Young's Modulus[9] is possible to relate the electrical resistance with the force applied to it.

Inside a load cell the strain gauges are placed following the Wheatstone bridge pattern:

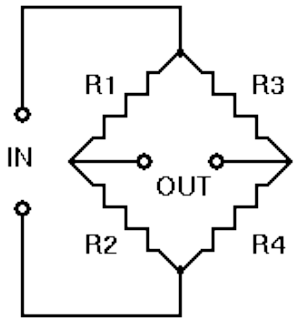


Figure 19 Wheatstone Bridge

Where Vin is a known constant voltage, and the resulting Vout is measured. If R1/R2 = R3/R4 then Vout is 0, but if there is a change to the value of one of the resistors, Vout will have a resulting change that can be measured and is governed by the following equation using ohms law:

$$Vout = \left[\left(\frac{R3}{R3+R4}\right) - \frac{R2}{R1+R2}\right] * Vin \quad (10)$$

By replacing one of the resistors in a Wheatstone bridge with a strain gauge, we can easily measure the change in Vout and use that to assess the force applied.

FUTEK LSB200

The load cell chose in this project is the model LSB200 from the manufacturer FLUTEK, this load cell was chosen by three main reasons:

- Its small size (17.5x19.1mm) allows us to place it easily in the small Gough-Steward platform.
- Its design its made for apply the force in the same axis as the mounting support, this is very important as normal load cell would have two axis one for the application of the force and other for the support two the reference, which would make necessary to add external hardware as shown in Figure 20.

[9] Young's modulus is a mechanical property that measures the stiffness of a solid material. It defines the relationship between stress and strain.

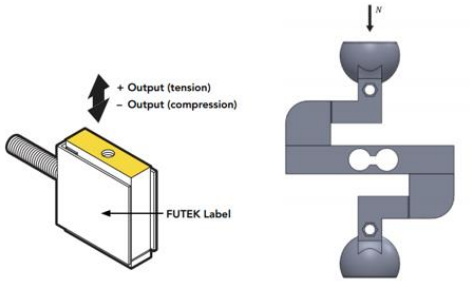


Figure 20 Comparison FLUTEK Load Cell vs Common Load cell

- Its low Hysteresis[10], Nonlinearity [11]and Nonrepeatability[12] ensures the precision of the system.

The main characteristics of the load cell are:

PERFORMANCE	
Nonlinearity ($\mathcal{E}_L$)	±0.1% of RO
Hysteresis ($\mathcal{E}_H$)	±0.1% of RO
Nonrepeatability ($\mathcal{E}_R$)	±0.05% of RO
ELECTRICAL	
Rated Output (RO)	2mV/V
Excitation (VDC or VAC)	10V max
Bridge Resistance	350 Ω
MECHANICAL	
Safe Overload	1000% of RO
Lb (L)	25lb
TEMPERATURE	
Temperature Shift Zero ($\mathcal{E}_Z$)	0.018% of RO/ºC
Temperature Shift Span ($\mathcal{E}_S$)	0.036% of Load/ºC

Table 3 FLUTEK LSB200 characteristics

From Table 3 we can calculate an approximation por the load cell total accuracy with the following equation:

$$\mathcal{E} < \sqrt{\mathcal{E}_L^2 + \mathcal{E}_H^2 + \mathcal{E}_R^2 + \left(\frac{\mathcal{E}_Z * L * N}{W_1} * t\right)^2 + (\mathcal{E}_s * t)^2} \quad (11)$$

Where N is the number of load cells to be used, W_1 is the maximum load to be measured and t the temperature variation range. Being N = 1 and supposing W_1 = 2Kg and t = 20ºC, then the total measurement accuracy of the load cell is:

$$\mathcal{E} < \sqrt{0.1^2 + 0.1^2 + 0.05^2 + \left(\frac{0.018 * 11.339 * 1}{3} * 20\right)^2 + (0.036 * 20)^2} \quad (12)$$

$$\mathcal{E} < \pm 1.5467\%$$

[10] Hysteresis: The maximum difference between the transducer output for the same load. One obtained by rising the load from zero and the other by decreasing the load from the RO.
[11] Nonlinearity: The maximum Deviation of the Calibration Curve from a straight line between 0 and RO.
[12] Nonrepeatability: The maximum difference between transducer output readings for repeated loadings.

3.2.2 Instrumentation amplifier

An instrumentation amplifier is a very popular precision differential voltage-amplifier that is optimized for operating in an environment hostile to get a precision measurement. One of its main features is that it does not require any input impedance matching as it is build including two input buffers.

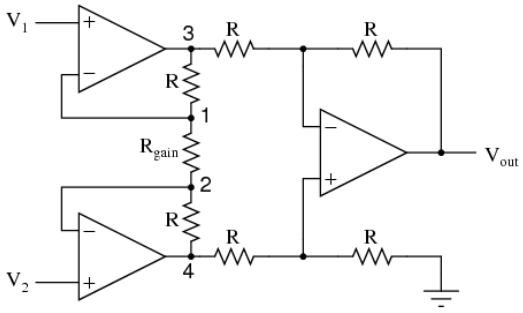


Figure 21 Instrumentation amplifier internal structure

General characteristic of differential amplifier includes; very low DC offset, low drift, low noise, very high open-loop gain, very high common-mode rejection ratio, and very high input impedances. Those characteristics are perfect for our application where the weak differential signal provided by the Load Cell needs to be highly amplified. Thanks to the high open-loop gain we can achieve almost any desired gain by adding a feedback circuit to it. The low DC offset and the low drift of the In-Amp result in a very precise amplification by reducing the error, nevertheless, in Chapter 3.3 the required hardware developed to reduce these errors will be discussed.

The real world is characterized by derivations from the ideal; temperature fluctuates, electrical noise exists, and voltage drops appears by current flow through the resistance of leads. Furthermore, real transducers rarely show zero output impedance or infinite input impedance, included, leaked or coupled electrical interference (noise) is always present to some extent. Instrumentation amplifiers are suited for application that requires to measure small voltage with high accuracy with minimal influence from noise. That's why choosing an In-Amp to this job was the best option.

MCP6N16

The instrumentation amplifier used in this project is the MCP6N16 form the manufacturer Microchip, this is an instrumentation amplifier based around an indirect current feedback topology consisting of three amplifiers: two matched transconductance amplifiers that convert voltage to current and one integrator amplifier that converts current to voltage.

This instrumentation amplifier has been chosen for presenting some very interesting features for this project, such as a high PSSR allows it to reject all the noise present in the power supply of the amplifier, as well as a high CMMR that allows to reject any common voltage in both differential inputs, as will be the case of any coupled source of noise. At the same time, the low drifts in temperature and time make it a very reliable option with very low offset.

At the same time, with the adequate resistances, the InAmp shows a practically imperceptible gain error. The features extracted from the most important datasheet are shown below.

Vos = 17µV	TC_1 = 60nV/ºC
PSSR = 128dB	TC_2 = 69pV/ºC
Δol = 137dB	I_B = 100pA
CMRR = 130dB	Ios = 800pA
$CMRR_2$ = 133dB	
g_E = ± 0.02	

Table 4 MCP6N16 characteristics

3.2.3 Load cell supply

Even though in this section the power supply components are not explained as not being considered relevant enough, the one responsible of powering the Load Cell needs to be explained deeply because its relevance.

The most important feature of this power supply is the noise it produces. Since this source powers the load cells, the noise in its output will be amplified by the InAmp. It is well known that this presents a very high CMRR, even so, it is not infinite and therefore it is necessary to ensure that the supply noise is low enough. Therefore, a low dropout linear regulator has been chosen as the power supply for this application. This decision has been made on the basis that this component has a high PSRR at the same time as an extremely low noise of 13.2µV RMS in addition to being able to stabilize with only a few components such as ceramic capacitors.

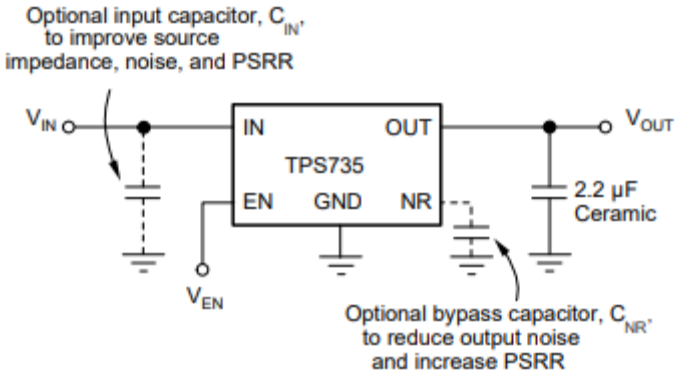


Figure 22 Load Cell Supply Typical Application

3.3 CIRCUIT DESIGN

As explained in chapter once the evaluation board provided by the manufacturer has been used in order to establish a first contact with the main component of the circuit, the instrumentation amplifier, we will proceed to design the main circuit. It is important to highlight that the design explained here is only the final version, but a lot of previous iterations were needed to achieve it.

3.3.1 Amplification

Next step is to design a circuitry that can amplify the weak differential signal provided by the load cell.

The first assumption of the circuit is the following one:

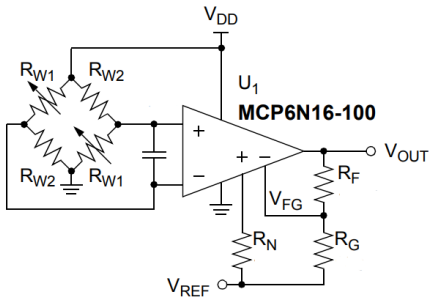


Figure 23 Amplification Circuit

As the application needs to measure either positive and negative forces the design criteria to choose the gain has been done considering that when no force is applied the signal should be centred at a value of 1,65v. Therefore:

$$Vout_{max} = 3.3V \quad V_{REF} = 1.65V \quad Vcc = 4.5V \quad RO = 2mV/V$$

At this point we need to define which is the maximum force the Load Cell will have to deal with, as at the moment of the design it was not known, an assumption of a weight of ±2Kg was done.

Then we can calculate the differential output of the Load Cell at this load:

$$\Delta V_{LC} = RO \cdot Vcc = \frac{2mV}{V} \cdot 4.5V = 9mV \quad at\ 25lb \tag{13}$$

$$\Delta V_{LC} = \frac{9mV}{25lb} \cdot 4lb = 1.44mV \quad at\ 4lb \approx 1.814Kg \tag{14}$$

Now we can build the equation that represents the behaviour of the circuit:

$$Vout_{max} = \Delta V_{LC} \cdot G + V_{REF} \tag{15}$$

Where we can find our desired gain.

$$G = 1074{,}218$$

Now we can look at the different values of the resistors to obtain the desired gain, the gain in the circuit from Figure 23 can be obtained from the following equation:

$$G = 1 + \frac{R_F}{R_G} \tag{16}$$

Then, the following values for the resistors were choose:

$$R_F = 51.7K\Omega$$
$$R_G = 47.5\Omega$$

Which ends providing the circuit a gain of $G = 1089.42$ which supposes a maximum force of 1.78Kg at each load cell. Then the R_N resistor was defined considering the following condition:

$$R_N = R_F || R_G = 47.5\Omega \quad (17)$$

As it will reduce the current bias of the circuit by stabilizing the load.

Gain Error

In order to analyse how sensible the error of the circuit will be, the followings equations were studied:

$$V_o = \left(1 + \frac{R_F}{R_G}\right) \cdot \Delta V_{LC} + V_{REF} \quad (18)$$

$$V_o' = \left(1 + \frac{R_F(1+\alpha)}{R_G(1-\alpha)}\right) \cdot \Delta V_{LC} + V_{REF} \quad (19)$$

$$e(V_o) = V_o' - V_o = \left[\left(1 + \frac{R_F(1+\alpha)}{R_G(1-\alpha)}\right) - \left(1 + \frac{R_F}{R_G}\right)\right] \cdot \Delta V_{LC} = \frac{-2R_F\alpha}{R_G(\alpha-1)} \cdot \Delta V_{LC} \quad (20)$$

Then by assuming that we are using really high-quality resistors with a tolerance of 0.01% the absolute error gives us a value of $e(V_o) = \pm 0.3344mV$. The relative error can be calculated as:

$$e_r(V_o) = \frac{e(V_o)}{V_o} = 0.01006\% \quad (21)$$

As we can see the gain error is absolutely small which is a very good and desired characteristic.

Now, if we consider the intrinsic gain error of the In-Amp then the total error is:

$$e_{In-Amp}(V_o) = ((G + g_E) \cdot \Delta V_{LC}) - (G \cdot \Delta V_{LC}) = 28.8\mu V \quad (22)$$

$$e_{TOTAL}(V_o) = e_{In-Amp}(V_o) + e(V_o) = 0.3632mV \quad (23)$$

Offset Error

Once the gain error has been calculated we can proceed to calculate the offset error. To do it the values from Table 4 extracted from the datasheet of the *MCP6N16* are used. In order to proceed to do the calculations the following assumptions were made:

$\Delta V_{DD} = 5v \pm 5\% = 0.25v$ $\Delta T_A = \pm 20^oC$
$\Delta VRef = 1.65 \pm 2\% = 0.033v$

Then we can proceed to calculate the increment of the common mode voltage:

$$\Delta V_{CM} = \left(I_B + {}^{I_{OS}}/_{2}\right) \cdot R = 0.1875\mu V \quad (24)$$

Where R = 375Ω as is the resistance of the Load Cell plus the filter resistance (assuming it is 200 Ω). Now we can finally calculate the global offset voltage:

$$V_E = V_{OS} + \frac{\Delta V_{DD}}{PSSR} + \frac{\Delta V_{CM}}{CMRR} + \frac{\Delta VRef}{CMRR_2} + \Delta T_A \cdot TC_1 + \Delta T_A{}^2 \cdot TC_2 = 18.67\mu V \quad (25)$$

Which at the output gives us a value of:

$$\Delta V_{OS} = V_E \cdot G = \pm 20{,}5mV \quad (26)$$

Desired Output vs Real Output

Once we have calculated the error in the gain and in the offset of our amplification we can proceed to calculate the full-scale error:

$$FS_{Error} = G_{Error} + O_{Error} = \pm 20.8344mV \quad (27)$$

From this result we can see that the offset error is much more relevant than the gain one, therefore, we will focus in a very effective way of removing it.

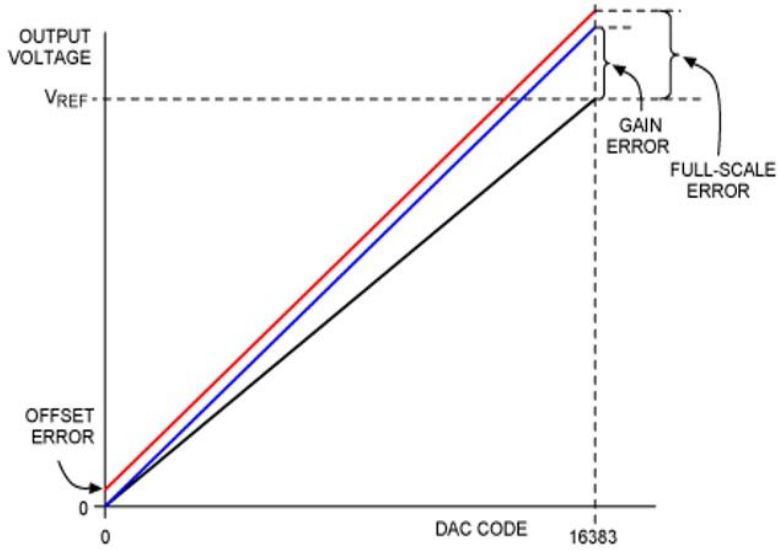


Figure 24 Real Gain Curve

3.3.2 Filtering

After amplification stage of signal conditioning, the signal must be filtered and optimized for ADC to read. The objective of adding filters to the circuit is to remove the intrinsic noise that will appear in almost any circuit and the possible coupled noise from the digital part of the circuit. At this point a second assumption needs to be done, as we do not know in which range of the frequency spectrum the useful information will be, a cutting frequency of 100Hz has been set.

Input filter

The first filter in the circuit is the input filter, this filter is responsible of filtering the noise provided by the voltage supply of the Load Cell or the coupled one at the long wires of the Load Cell.

The values of the resistor were chosen to be as small as possible in order to reduce the amount of noise they provide to the input of the amplification due to Johnson effect[13]. As the signal provided from the load cell is a differential signal, the decision of using a differential filter was clear as it provides higher linearity, immunity to common-mode interference signals, etc.

When designing this filter, the value of C2 was limited to be at least ten times larger than C1 to reduce the effects of the time constant mismatch and improve its performance.

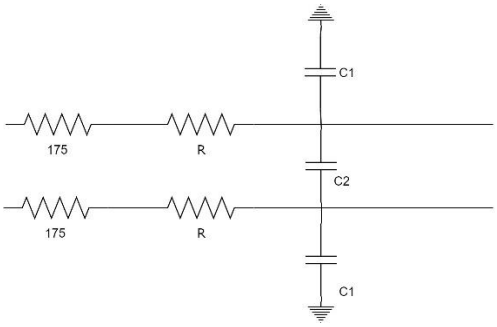


Figure 25 Differential Input Filter

The following values where chosen in order to obtain a cutting frequency of 404.2Hz.

- R = 200Ω
- C_1 = 0.1µF
- C_2 = 1 µF

Note that in order to find the actual cutting frequency the resistance of the Load cell must be considered. Therefore, the cutting frequency can be calculated as it follows:

$$f_{c1} = \frac{1}{2\pi \cdot 2 \cdot (R+175) \cdot (C_2 + \frac{C_1}{2})} \quad (28)$$

Feedback filter

The second filter is the feedback filter, this filter is placed in the feedback stage of the amplification, this filter is responsible of rejecting high frequency harmonics that might have skipped the first filter or appeared inside the In-Amp.

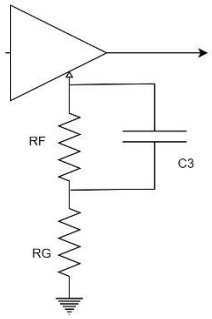


Figure 26 Feedback filter

13

From Figure 26 we can see that the filter uses the RF resistor predefined by the amplification stage, thus, the cutting frequency of this filter only can be used by tweaking the capacitance of C_3. Therefore, the cutting frequency of this low pass filter can be calculated using equation 29.

$$f_{c2} = \frac{1}{2\pi \cdot 51.7K \cdot C_3} \tag{29}$$

Where a value of C_3=3.9nF has been chose to obtain a cutting frequency of 789.34Hz.

Output filter

The last filter is the filter placed at the output which has been included there only to ensure the proper filtering process and to create a third order filter together with the other two filters.

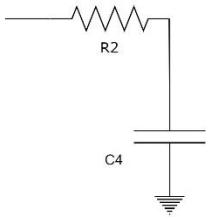


Figure 27 Output Filter

Using equation 30 the values of R_2 = 10K & C_4=0.01µF where chosen to obtain a cutting frequency of 1591.51Hz.

$$f_{c3} = \frac{1}{2\pi \cdot R_2 \cdot C_4} \tag{30}$$

Total Frequency response

At this moment we can combine all the parts in order to create a first approach of the circuit.

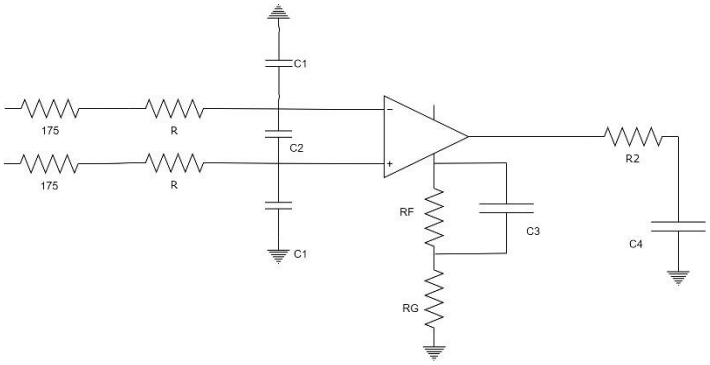


Figure 28 Final Circuit Approach

As said before the combination of the three filters placed in this circuit makes it to behave like a third order filter. The total cutting frequency can be estimated as it follows:

$$f_c = \frac{1}{\frac{1}{f_{c1}} + \frac{1}{f_{c2}} + \frac{1}{f_{c3}}} \tag{31}$$

Which gives us a cutting frequency of 228Hz where the response of the amplification it attenuates -9dB with a slope of -60dB/dec.

3.3.3 Final circuit

Finally, the circuit can be completed by adding the required extra components.

One of the main components the circuit needs is a voltage reference of 1.65v in order to offset the signal when no force is applied, otherwise, as this circuit uses a rail to rail In-Amp where only a positive supply is fed, it wold be impossible to measure negative forces.

Taking advantage of the fact that a reference signal of 1.65 volts has to be introduced, we can use the same hardware to vary this signal and thus compensate for the possible offset. Next we will see how this can be carried out.

Offset compensation

As seen in section 3.3.1 the offset error us much bigger than the gain error. Therefore, in the process of the different design iterations some approaches were taken in consideration in order to solve it.

The first proposal was to use of a digital potentiometer followed by operational amplifier in a buffer configuration, the digital potentiometer allow us to create a voltage divider centered at the desired voltage where we can choose the size of the required steps.

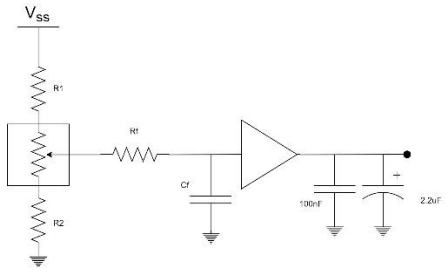


Figure 29 Digital Pot for Offset Compensation

Although analysing the results obtained, the idea of using a digital potentiometer seems to be a viable solution for the problem that has been proposed, nevertheless, it has been decided not to use it. The fact that the system requires many components and that there is a risk of not being able to compensate the offset when the load cells are too preloaded made us look for another solution.

The next option is to opt for a very common method and used everywhere as it is a DAC. These devices allow generating an analogic voltage based on a reference that can be varied step by step, just like the previous method. The benefit is that it will allow us to generate the voltage from 0v to the reference voltage and that the necessary components will be a lot less, but against, each step will be much bigger than in the previous case therefore we will not have as much resolution as before.

The proposed solution consists of a 12-bit DAC connected to a reference voltage of 1.8 volts, which has been chosen for its low drift in temperature, its low current consumption and its accuracy of 0.05%. To control the DAC we have chosen a model that allows to be interacted via the I2C communication protocol, likewise the other components that will be chosen in this circuit will also be connected to the same I2C communication bus facilitating the task of managing the peripherals of the circuit.

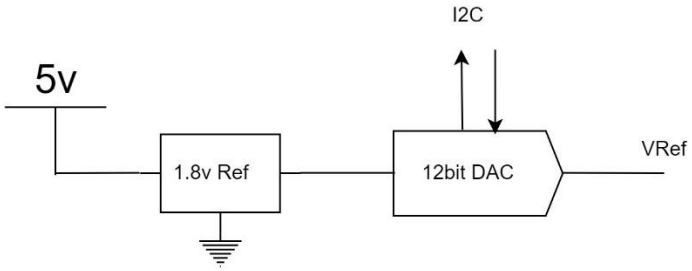


Figure 30 Offset Compensation DAC

One of the risks that are assumed when components of this type are introduced into the circuit is the coupling of the noise. Therefore, it has been decided to take into account the characteristics and the pertinent noise graphics provided by the manufacturer of the components.

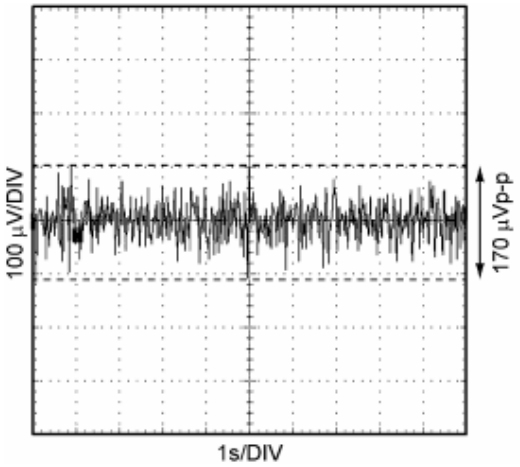


Figure 31 Voltage Reference Noise

Since the manufacturer of the DAC does not provide the relevant graphs for noise analysis, we can only analyse the reference voltage. From which we can extract that this reference voltage is not prone to introduce noise by itself considering the few microvolts of negligible noise. Therefore, everything will depend on the purity of the power signal that we introduce and how the digital part of the analogic DAC is isolated

Gain Calibration DACs

On the other hand, the profit error is still present and despite being very small it can affect our readings. Therefore, in this project, we have opted to find an alternative method to be able to calibrate the said gain error and thus try to approximate the circuit to its ideal performance. The solution that has been found has been to use two 16-bit DACs to generate a differential signal small enough and at the same time known to be able to enter it in the In-Amp input and thus observe the output and quantify how far it is from the expected.

EEPROM memory

Finally, in order to allow the microcontroller to save the necessary values to emulate a slave device of the Dynamixel bus, it is necessary to provide the circuit with an EPROM memory. This memory must also be accessible via I^2C. Therefore, we have looked for one that was of a reduced budget and that it had enough memory to be able to store all the required register tables.

The complete schematic of the circuit designed in Orcad can be found in Appendix C, as well as the Bill of Materials in Appendix B.

3.3.4 Simulation

Once the theoretical calculations on the performance of the circuit have been established, the complete circuit has been designed in Orcad in order to carry out the relevant simulations to verify that the calculations were correct. Next, we will see and discuss the results obtained in the simulations.

The first simulation that is performed is in order to verify that the circuit amplifies as expected, to do it a sinusoidal differential signal is introduced emulating the maximum and minimum force of the load cells, at the same time this signal is of a frequency of 1 Hz to avoid being attenuated by the action of the filters.

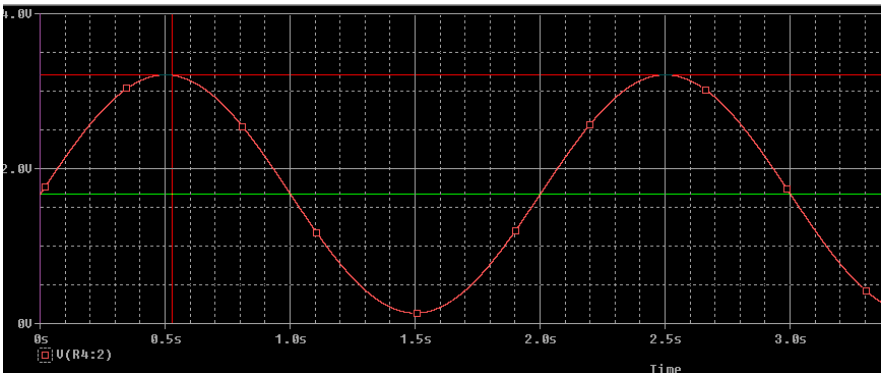


Figure 32 Gain Simulation

As we can observe the signal resulting from the amplification is centered at 1.65 volts and reaches a maximum value of 3.3 which indicates that the calculations are correct. On the contrary, in the lower part it does not reach a value of 0 volts and this is due to an intrinsic characteristic of the instrumentation amplifier. Being the rail to rail type does not allow to reach values close to zero.

$$Minimum\ Output\ Voltage\ Swing = 6mV$$

The next simulation that is performed is the frequency response, in this we intend to observe that the slope of the circuit frequency response is -60 dB/dec and that the cutting point is located around 200Hz attenuating - 9 dB.

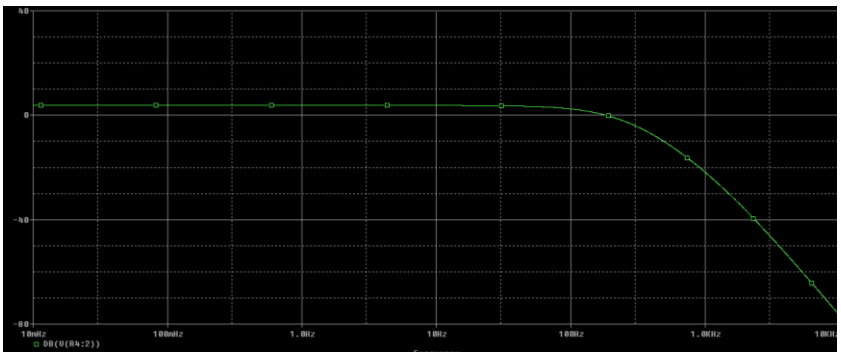


Figure 33 Bode Diagram Simulation

Observing the resulting Bode diagram, we can conclude that the analysis of the frequency response has also been satisfactory, obtaining the answer predicted by the mathematical calculations.

Finally, to check if there is enough PSSR in the amplifier to reject a majority of the noise located in its power supply, a noise source has been artificially introduced into the circuit to observe how it modifies the output of the amplification.

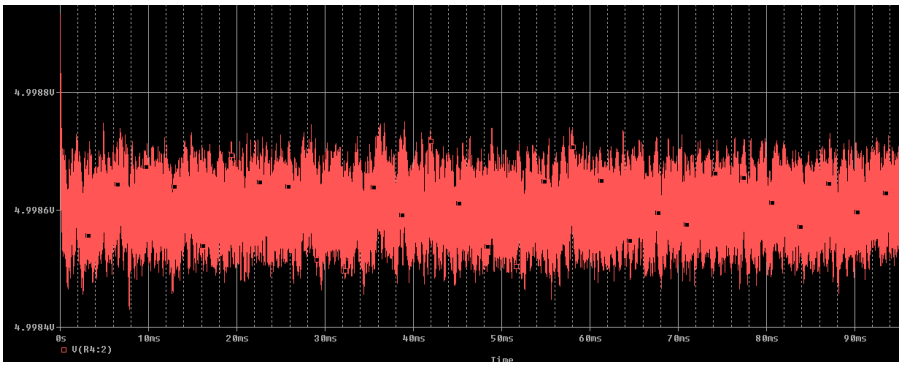


Figure 34 Noise meaning at the Output

3.4 PCB DESIGN

The printed circuit or PCB (Printed Circuit Board), is a surface consisting of paths or tracks of conductive material on a non-conductive base. The printed circuit is used to connect electrically, through the paths, and will mechanically support the set of electronic components.

A PCB can have the upper (Top) and lower (Bottom) layers to mount the

components and connections, and also can have internal layers where you can do the routing of the lines or use them as mass plans or feeding. In Figure 35 a two-layer PCB can be seen and in Figure 36 a 4-layer layout can be observed.

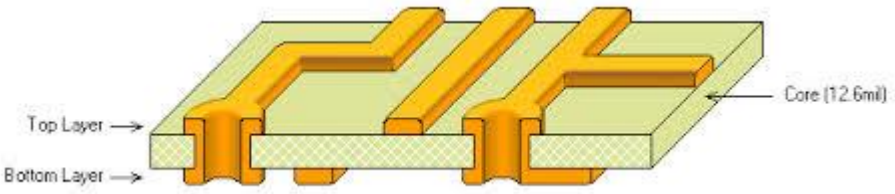


Figure 35 Two-Layer PCB

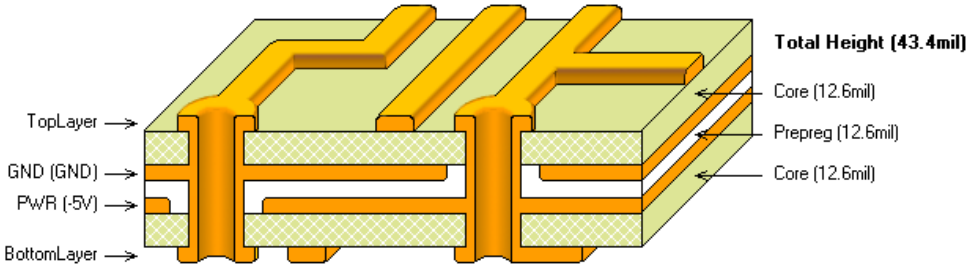


Figure 36 Four-Layer PCB

On the other hand, in the manufacturing process of the PCB, masks are used to delimit the connections or the layers of silkscreen printing. The dimensions of the pads and the lines may require more or less complex masks. Manufacturers classify PCBs based on the

difficulty of masks. Therefore, when smaller pads, lines or paths are required, more complex technologies are used, therefore a class of greater difficulty and also a greater cost.

Power and Ground can be made to reach the components by using transmission lines or using internal layers connected to planes. Normally the transmission lines are used in simple circuits with little component density or with a few different power supplies. On the other hand, if the density of components is high or there are many different power supplies, the internal layers are used using the planes where you can connect to the top or the bottom via tracks. The use of internal layers in a PCB increases the cost because it has a manufacturing process more complex than the transmission lines.

When choosing the components, it must be taken into account that there are two technologies, the through-holes technology (THT) or surface mounting (Surface Mounted Devices, SMD). The main difference is that the THT require holes that perforate the PCB and therefore occupy space both in the Top and in the Bottom, whereas the surface mountings only occupy space on either side. Therefore, with the components of surface mount can achieve a higher component density.

To perform the PCB it has been decided to use the maximum number of components in SMD technology and the minimum number of THT components, since not all can be found in surface mount. As the PCB of this circuit has to bundle with both analogic and digital circuitry, it is necessary to isolate the ground planes of the two parts of the circuits. For this reason, it will be necessary to have one of the layers dedicated exclusively to the ground plane and therefore the PCB has to be of 4 layers. Taking advantage of this, the fourth layer will be used to supply the power lines.

The method of separation of the ground planes allows to avoid returns of the flow of digital current to the analogic part and thus the coupling of noise in the analogic signal. This is obtained by means of the absolute separation of the two planes except at a relatively small point so an increase in impedance is achieved at that point, causing any return of current to be attenuated by that increase in impedance. In really sensitive circuits or with a very high precision, instead of using a high impedance point, we choose to place an inductance.

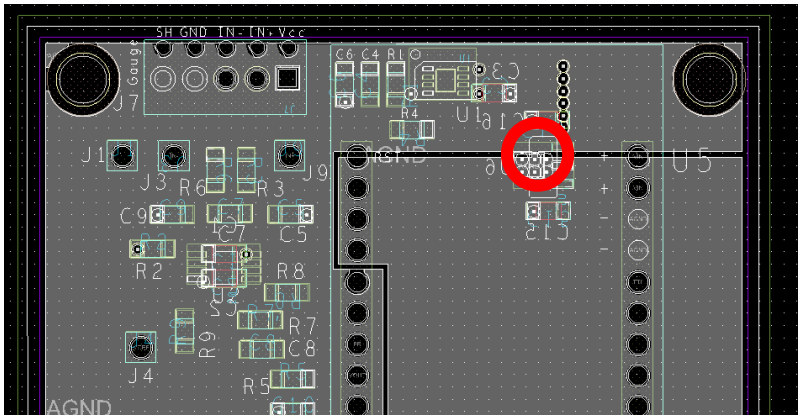


Figure 37 Separate Ground planes connexion

General criteria working with OrCAD:

First of all, the rules that the program must review are configured. These rules are the minimum and maximum size of the lines and tracks, the minimum distance between different components, lines and routes, and other types of rules.

Once configured, the program loads all the components that are found in the electrical diagrams and are distributed separating the analog and digital part. Then the lines connecting the different components are drawn. In order to draw the lines, some basic recommendations have been followed that can be observed below:

- Avoid stubs on the transmission lines. A stub is a transmission line connected to another line at one end and the other end being shorted or in open circuit. Stubs vary the impedances of the transmission lines.

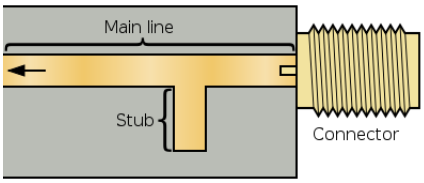


Figure 38 Stub Illustration

- Avoid crosstalk between two signals. The crosstalk occurs when two lines are close and a part of the signal from a line disrupts the other line. It is recommendesd that the transmission lines be separated between them centers four times its width.
- Try to make the lines as wide and short as possible. Long and thin lines produce a loss of tension at the other end.
- Power vias should be wider than signals. One standard width measurement of the power lines is 1 mm.
- Decoupling capacitors must be placed as close as possible of the power terminal and the IC which it filters. It's also preferable the use of SMD capacitors.
- Avoid making angles of 90º with the lines of transmission.

3.4.1 Assembly and Results

Once the design of the PCB has been completed, the generated gerbers are sent to an external manufacturer who is in charge of the production process of the PCBs. When the PCBs are received the components are assembled to them.

For this purpose, the soldering station found in the IRI laboratories has been used, as well as all the tools, such as weld flux, tin, pliers, magnifying glass, etc.

Figure 39 Soldering Station at the IRI

Once the PCBs have been assembled, you can proceed to verify that they work correct. For this, they are connected to a laboratory power supply in order to monitor the output with the oscilloscope. Then we verify that everything works correctly, one of the most interesting aspects to analyse in this circuit is the noise produced at the output, as well as the noise of the power supply to the load cells. Below are two graphs of said samples.

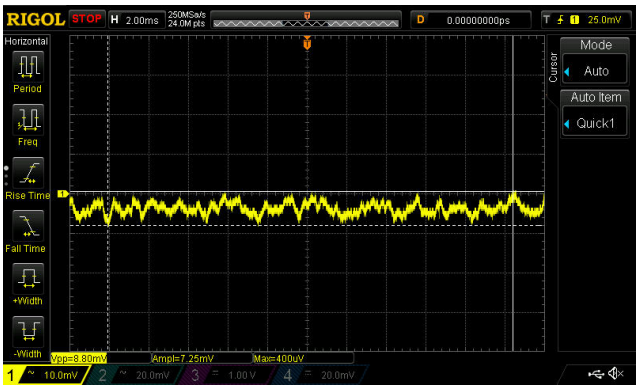


Figure 40 Output noise

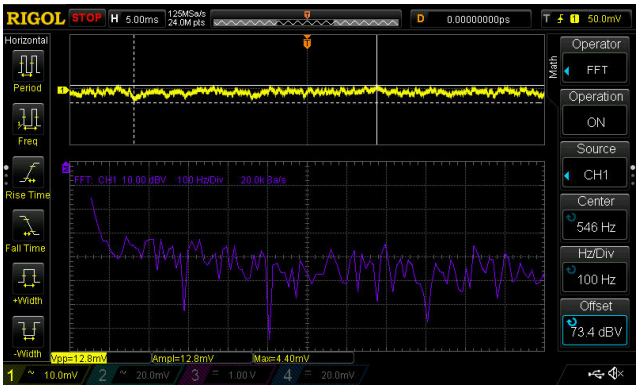


Figure 41 DFT Output noise

Analysing Figure 40 and Figure 41 we can see that the output of the amplification is equipped with a noise component of about 8 millivolts of amplitude at very low frequency.

Although a less noisy performance of the circuit was expected, the noise component is sufficiently low to be practically imperceptible, which is why the design of the circuit is thought to be quite satisfactory

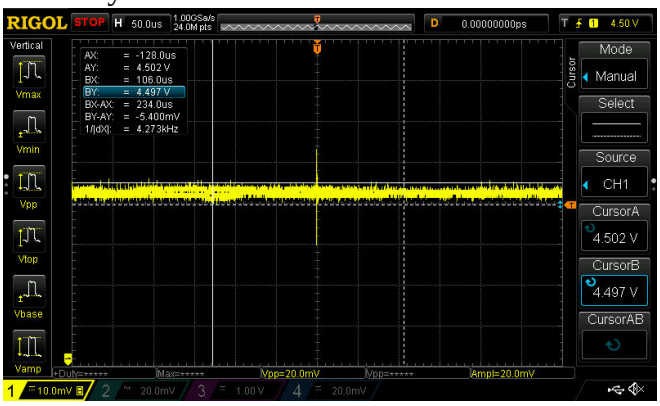


Figure 42 Load cell Power Supply

Finally, what was really surprising was the result obtained when measuring the noise in the voltage supply of the load cells. The component was chosen rigorously in order to obtain very low noise output, however the analysis done at the oscilloscope we have observed a noise amplitude of up to 5 millivolts which is quite disappointing compared to the characteristics that the manufacturer established. Therefore, it is believed that some parameter of the LDO stability was not chosen correctly.

4 APPLICATION

This chapter is splitted in the three parts, the two firsts combined result in the third which is the work of the project. The chapter addresses the problem of determining the point of application of the force and the angle of the actuators to move the platform to its desired position.

At this point of the thesis, it is assumed that all the communications between the different devices is already established.

4.1 ACTUATION

This section explains how the problem of moving the platform is mathematically treated according to the mechanics of the platform.

This section deals with all the algebraic work needed to solve the inverse kinematics specific to the robot used in this project. It explains step by step the definition of references of the robot and the obtaining of the analytical solutions for the angles of the actuators. Matlab software, along with the Robotics toolbox and the functions introduced in Chapter 2.3.3, are of great help to achieve this goal.

4.1.1 Calculation of the Solutions

Before start dealing with the needed calculations to solve the inverse kinematics, some abbreviations must be defined to compact the notation used:

- For the pure translation between two references linked by a vector in 3 dimensions:

$$Transl = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{32}$$

- For a pure rotation of any angle around the axes of the base:

$$RotX(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) & 0 \\ 0 & sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{33}$$

$$RotY(\varphi) = \begin{pmatrix} cos(\varphi) & 0 & sin(\varphi) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\varphi) & 0 & cos(\varphi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (34)$$

$$RotZ(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 & 0 \\ sin(\theta) & cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (35)$$

To reverse the direction of a transformation, simply invert the matrix that relates the reference systems.
The measures used in the following sections (L1,L2,..) have been extracted from the CAD design and can be found in the Appendix G.

Absolute frame reference

The first step when computing the inverse kinematics is to define a frame reference, this reference must be defined somewhere in order to express the remaining frames. The chosen point to place the reference is in the centre of the platform's base.

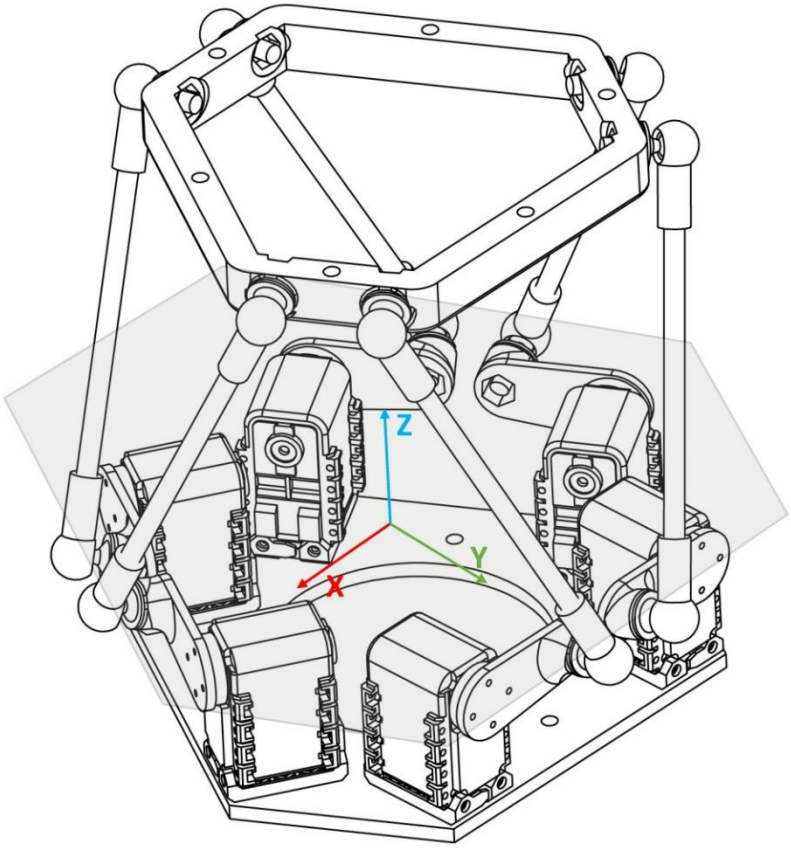


Figure 43 Absolute Reference Frame Position

The height of the absolute reference is chosen to be at the centre of the Dynamixel actuators rotor axis as it makes the calculations easier. This frame will receive the name of TBase.

Platform's reference frame

The next step is the definition of the transformation that takes the TBase reference to the final position.

This desired position depends on how the platform needs to be moved. In this case, for the application, it has been decided that the platform rotates around an axis parallel to the plane of the platform at rest and at a predefined distance H. Nevertheless, in this section we will consider all the possible movements.

Then, the position of the platform's reference can be expressed as:

$$TPlat = transl(\Delta x, \Delta y, \Delta z + H) \cdot RotX(\alpha) \cdot RotY(\varphi) \cdot RotZ(\theta) \quad (36)$$

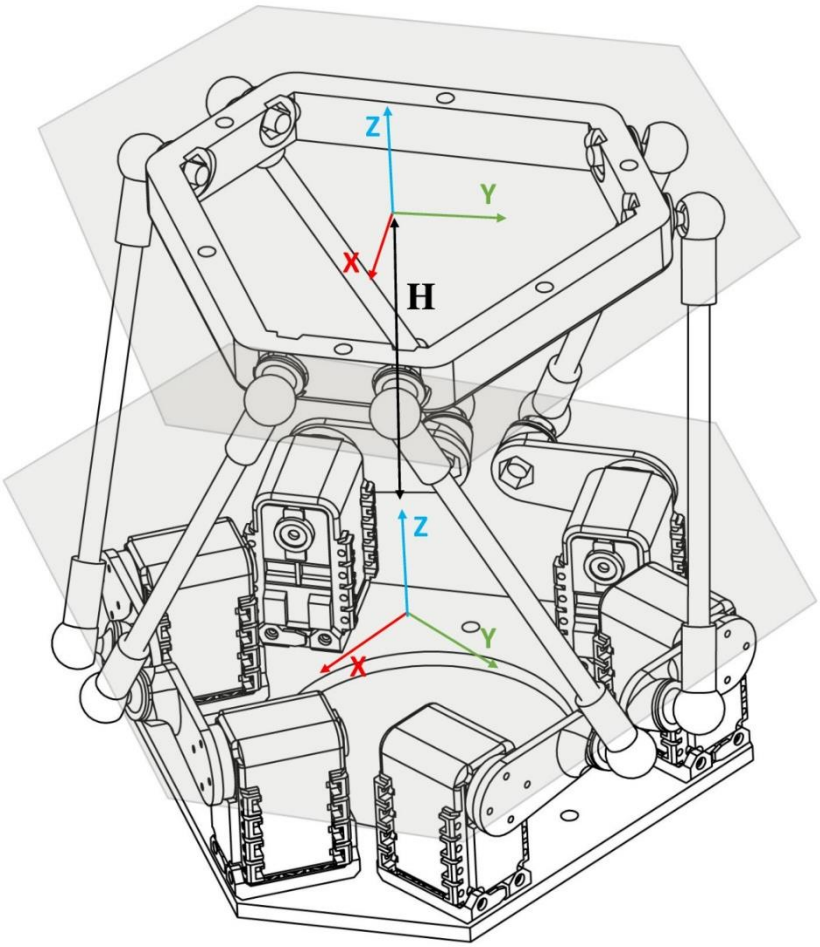


Figure 44 Platform Frame Reference

Frames of the joints with respect to the base

Once the two references are well defined one with respect to the other, in order to solve the kinematics calculous a frame must be placed at the end of each leg, the joint (top joint not the actuator). The result will be six matrices that will represent the position of each joint with respect to the Base.

$$TBase = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (37)$$

These references are obtained by applying a translation and a rotation to the platform reference TPlat, as this is also referenced to the base reference TBase.

$$TP_1 = TPlat \cdot Transl(L1, -L2, 0) \cdot RotZ\left(\pi/2\right) \quad (38)$$

$$TP_2 = TPlat \cdot RotZ\left(-2\pi/3\right) \cdot Transl(L1, L2, 0) \cdot RotZ\left(\pi/2\right) \quad (39)$$

$$TP_3 = TPlat \cdot RotZ\left(-2\pi/3\right) \cdot Transl(L1, -L2, 0) \cdot RotZ\left(\pi/2\right) \quad (40)$$

$$TP_4 = TPlat \cdot RotZ\left(2\pi/3\right) \cdot Transl(L1, L2, 0) \cdot RotZ\left(\pi/2\right) \quad (41)$$

$$TP_5 = TPlat \cdot RotZ\left(2\pi/3\right) \cdot Transl(L1, -L2, 0) \cdot RotZ\left(\pi/2\right) \quad (42)$$

$$TP_6 = TPlat \cdot Transl(L1, L2, 0) \cdot RotZ\left(\pi/2\right) \quad (43)$$

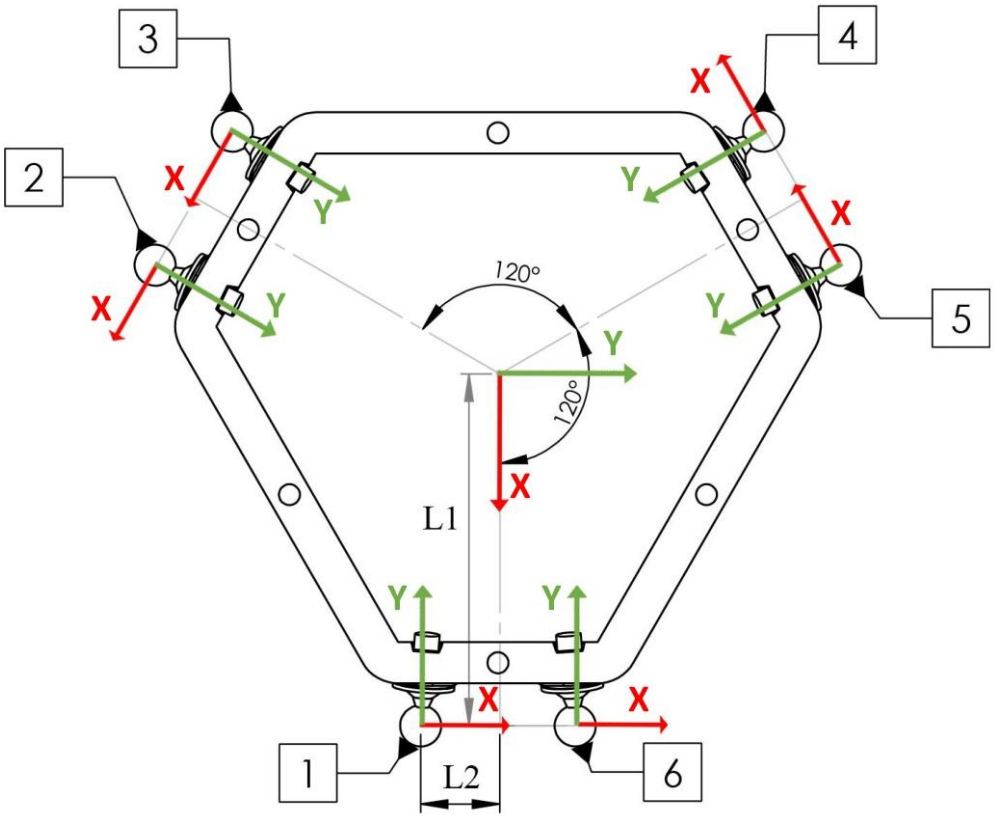


Figure 45 Joints' Frame Reference

Frames of the actuators with respect to the base

As well as done before with the joints, the reference frame needs to be translated from the centre to the rotor axis of the actuators. Notice that the TBr matrices will be defined to have the X axis coincident to the actuator arm at position 0, as it will make the calculations easier.

$$TB_1 = TBase \cdot Transl(L3, -L4, 0) \cdot RotZ\left(-5\pi/6\right) \quad (44)$$

$$TB_6 = TBase \cdot Transl(L3, L4, 0) \cdot RotZ\left(-\pi/6\right) \quad (45)$$

$$TB_2 = RotZ\left(-2\pi/3\right) \cdot TB_1 \quad (46)$$

$$TB_3 = RotZ\left(-2\pi/3\right) \cdot TB_6 \quad (47)$$

$$TB_4 = RotZ\left(-4\pi/3\right) \cdot TB_1 \quad (48)$$

$$TB_5 = RotZ\left(-4\pi/3\right) \cdot TB_6 \quad (49)$$

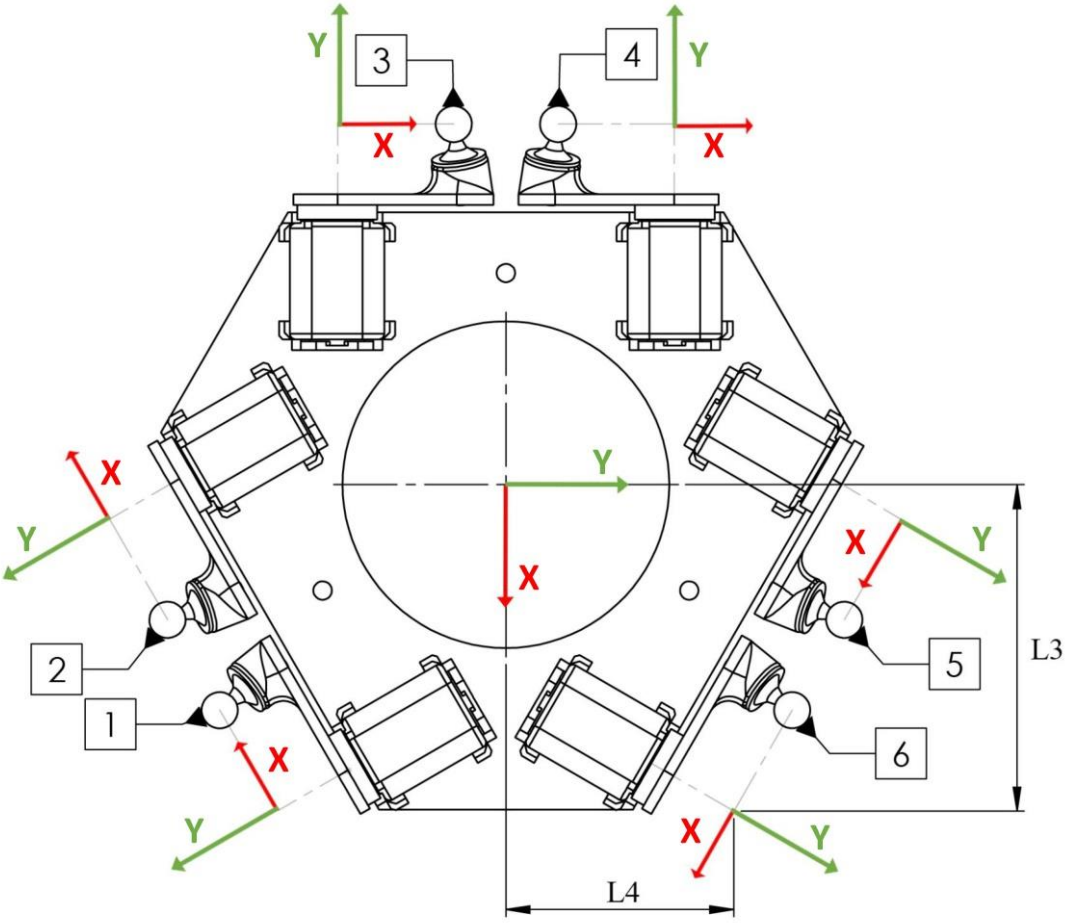


Figure 46 Actuators' Frame Reference

It is important to notice that the position of the axis is separated from the actual actuator arm, this is because the actual leg is connected to the spherical joint, which moves the action line, and, therefore, the reference.

Resolution of the equations

Finally, we can proceed to calculate the solutions. For each leg of the platform, the procedure is as follows:

First of all, the joints references need to be expressed at the same coordinates as the base, to do this it is necessary to go through the transformations that relate them. The subscript ***i*** indicates the number of the leg:

$$T_i = TB_i^{-1} \cdot TP_i \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix} \tag{50}$$

At this point the problem is reduced to find and to solve the equations that relate to the actuators' rotor position to the spherical joints' position. In this case the possible solutions correspond to the intersection of a sphere defined by the length of the arm of the actuator as the radius with the arm of the platform.

Then, in order to find the mathematical relationship, the manipulator must be analysed in more detail.

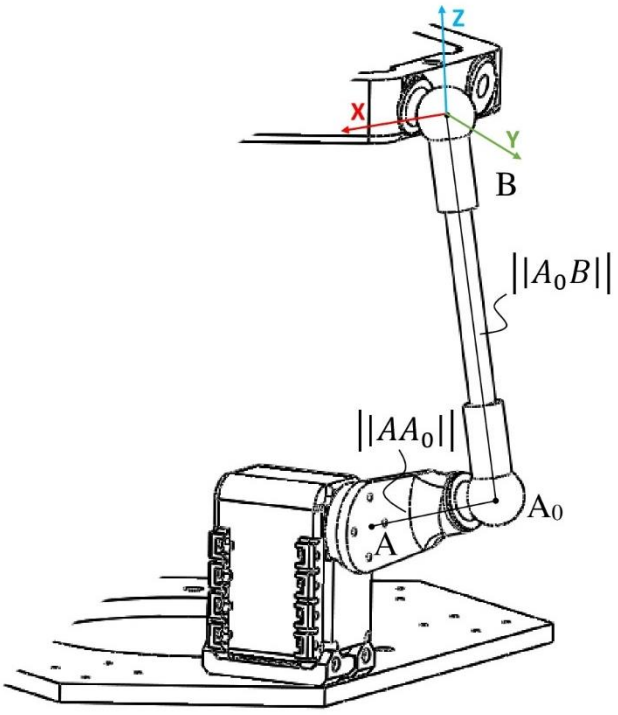


Figure 47 Actuator + leg mechanism

As it can be seen in Figure 47, the rotor movement is transmitted through the mechanism AA_0B. The problem of the inverse kinematics is to determine the angle between the mechanism AA_0 and the vector of the rotor frame placed in the initial position of the mechanism.

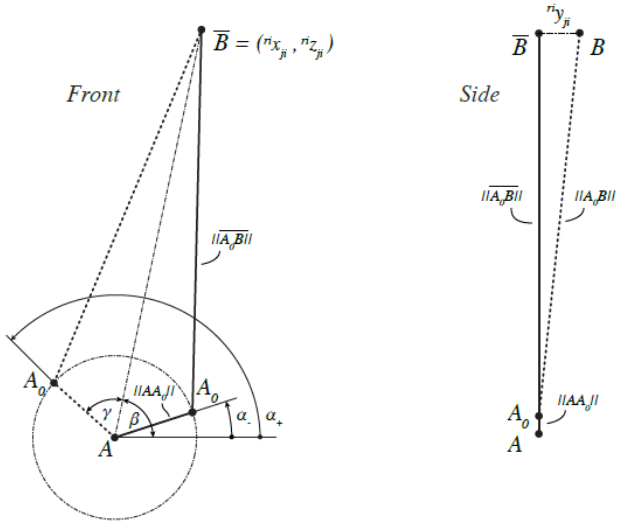


Figure 48 Mechanism Servo Arm + Leg

As $\left||A_0B|\right|$ is the length of the platform leg $||\overline{A_0B}||$ can be found as:

$$\left||\overline{A_0B}||\right| = \sqrt{\left||A_0B|\right|^2 + y^2} \tag{51}$$

Then β can be found as:

$$\beta = tan^{-1}(\frac{Z}{X}) \tag{52}$$

Knowing that $\left||A_0B|\right|$ is the length of the arm of the actuators we can calculate the value of Υ applying the cosine theorem:

$$\Upsilon = \cos\left(\frac{\left||AA_0|\right|^2+x^2+z^2-\left||A_0B|\right|^2}{2\cdot\left||AA_0|\right|\cdot\sqrt{x^2\cdot z^2}}\right) \tag{53}$$

Finally, the actual value α can be obtain for two solutions:

$$\alpha = \beta \pm \Upsilon \tag{54}$$

α can have two possible solutions as the point A_0 lies on a sphere where the constrain of finding a point from B of length $\left||\overline{A_0B}|\right|$ will always intersect in two points. The whole inverse kinematics admits generally 2^6 = 64 solutions for each platform configuration.

Therefore, one of the solutions is imposed in each actuator:

$$\alpha_1 = \beta + \Upsilon \tag{55}$$
$$\alpha_2 = \beta - \Upsilon \tag{56}$$
$$\alpha_3 = \beta + \Upsilon \tag{57}$$
$$\alpha_4 = \beta - \Upsilon \tag{58}$$
$$\alpha_5 = \beta + \Upsilon \tag{59}$$
$$\alpha_6 = \beta - \Upsilon \tag{60}$$

Saturation values

Unlike the resolution of the inverse kinematics problem, in these methods there is no way to know if an impossible configuration for the platform is being executed. This leads to find the joint values from which the platform movement (about 18º) begins to be blocked by the joint restrictions. In order to keep the manipulators away from dangerous configurations, these positions define the saturation values for the actuators, which will be also implemented in the code.

4.1.2 Simulator

As explained in Chapter Matlab Robotic toolbox2.3.3, Matlab and the robotics toolbox have been an essential tool to verify all the calculous done in this chapter. Therefore, a simulator that verifies the calculus done in the previous sections was developed.

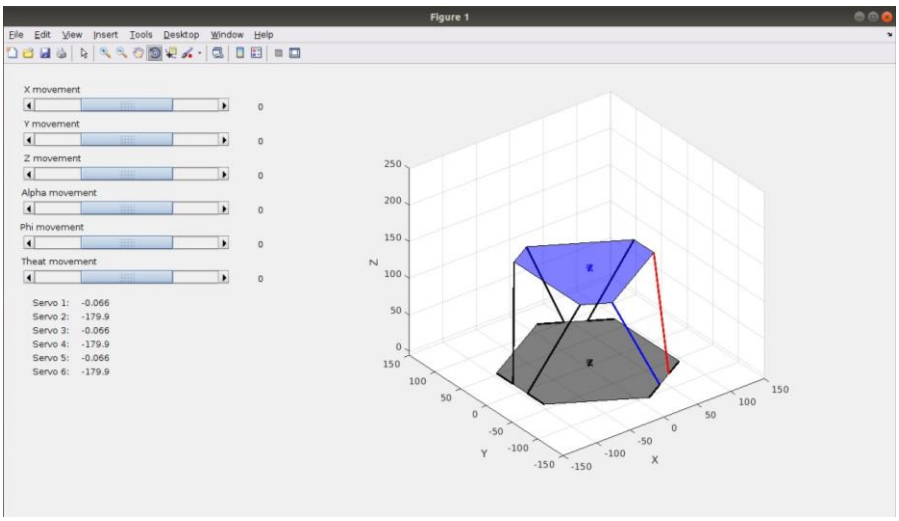


Figure 49 Matlab kinematics simulator

This simulator consists of a window that shows the platform in a 3D representation, six sliders which control the position three for the translations and three for the rotations and the angle of the servo arm. The code of this simulator can be found in Appendix I.

4.2 FORCE/TORQUE SENSOR

This section explains how the problem of finding the point of application of a force in the plate of the platform is addressed. This section is based on the work done in [1].

The weight of the platform it is not reflected in the equations since the program itself is calibrated so that load cells give value of 0 when the platform has no load.

Point of application of outer force

Once the external force wrench has been calculated on the platform, this information can be used to deduce others that may be useful. As we will see below, assuming that the external force and torques coincide in direction, from the resulting wrench we can deduce:

- External force vector
- External torque vector
- Action line of the external force

Therefore, once we have the action line of the force we can calculate its intersection with the platform and deduce the point of application of the force on the platform.

There is a special case in which, despite having an outer torque on the platform, we can calculate the force line of outer force. This is where the torque has the same direction as the force applied on the platform. The torque of the forces and torques outside the platform can be represented as:

$$f_{plat} = \begin{pmatrix} \vec{F} \\ \vec{M} \end{pmatrix} \tag{61}$$

Together with equation (9) can be expressed as:

$$\begin{pmatrix} \vec{F} \\ \vec{M} \end{pmatrix} = - \begin{pmatrix} \sum_{i=1}^{6} f_i \, \vec{e}_i \\ \sum_{i=1}^{6} f_i \, \vec{e}_i \times \vec{r}_i \end{pmatrix} \tag{62}$$

Where $\vec{F}$ is the sum of external forces and $\vec{M}$ represents the sum of all external moments with respect to the reference point. This sum of moments includes both the pure pairs applied on the platform and the moments realized by the different forces. In the case discussed above, in which only an external force acts and in which the only outer torque has the same direction of force.

$$\vec{M} = \overrightarrow{OO'} \times \vec{F} + \vec{\Gamma} \tag{63}$$

Where O is the reference point, O' a point pertaining to the acting force line of the force and $\vec{\Gamma}$ is the outer torque that has the same direction as the force. According to the conditions we have imposed by the pure outer torque: $\vec{\Gamma} = k\vec{F}$ with $k \in \mathbb{R}$, and, if we consider that $\overrightarrow{OO'} = (x, y, z)^{\top}$, the equation (63) remains:

$$\vec{M} = (x, y, z)^{\top} \times \vec{F} + \overrightarrow{kF} \tag{64}$$

To determine totally x, y, z and k we need an additional equation which fixes the position of O' along the acting line of the force. This equation will be the one that determines the cutting point between the external force acting line and the sensor platform.

$$z = 0 \tag{65}$$

Combining the equations (64) and (65) we obtain:

$$\begin{pmatrix} M_1 \\ M_1 \\ M_1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & F_3 & -F_2 & F_1 \\ -F_3 & 0 & F_1 & F_2 \\ F_2 & -F_1 & 0 & F_3 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ 0 \\ k \end{pmatrix} \tag{66}$$

where F_i and M_i represent the i[th] component of the $\vec{F}$ and $\vec{M}$ vectors.

The resolution of this linear system results in the next expression:

$$\begin{pmatrix} x \\ y \\ z \\ k \end{pmatrix} = \begin{pmatrix} \frac{F_1F_2M_1 - F_1^2M_2 - F_3^2M_2 + F_2F_3M_3}{F_3(F_1^2 + F_2^2 + F_3^2)} \\ \frac{F_2^2M_1 + F_3^2M_1 - F_1F_2M_2 - F_1F_3M_3}{F_3(F_1^2 + F_2^2 + F_3^2)} \\ 0 \\ \frac{F_1M_1 + F_2M_2 + F_3M_3}{F_1^2 + F_2^2 + F_3^2} \end{pmatrix} \quad (67)$$

Matlab verification

As well as in the previous section in order to verify that the mathematic study of the force/torque sensor has been correctly done, a Matlab code has been develop. This code reads a ".txt" file which contains the readings of the forces through the six legs, then it will compute and plot the X & Y coordinates of the applied force.
This code can be found in Appendix H and will be used in Chapter 6.1 to verify the results.

4.3 COMPLIANT ACTUATOR

Once the point of application and the amount of force is known the desired position will be compute considering to predefined values:

- X axis Spring Resistance: K_X [º/mN*mm]
- Y axis Spring Resistance: K_Y [º/mN*mm]

These values will define the amount of degrees each axis will rotate according to the torque applied perpendicularly to the axis, this one, being referenced to the centre of the platform.

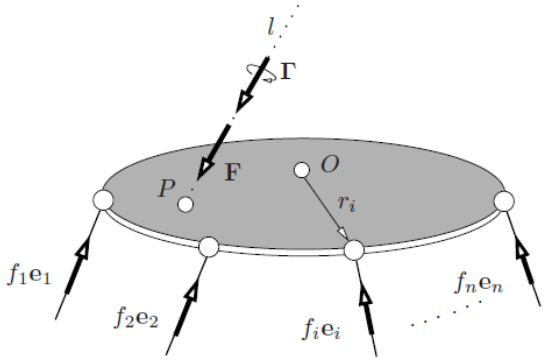


Figure 50 Force application relative to origin

In order to calculate the torque, the following equations are used:

$$\Gamma_X = P_1 \cdot F \quad (68)$$
$$\Gamma_Y = P_2 \cdot F \quad (69)$$

*Note that this torque is expressed in [mN·mm] for more accuracy.

Then the desired output is computed by:

$$\varphi = \Gamma_X \cdot K_X \quad (70)$$
$$\alpha = \Gamma_Y \cdot K_Y \quad (71)$$

We have to note that after moving the platform the coordinates of the $\vec{e}_i$ vector will change from its original value, so, we will need to recalculate its value.

First, as we don't know the reference of A_0 we need to calculate it:

$$TA_{01} = TB_1 \cdot RotZ(\alpha_1) \cdot Transl(ServoArm, 0, 0) \quad (72)$$
$$TA_{02} = TB_2 \cdot RotZ(\alpha_2) \cdot Transl(-ServoArm, 0, 0) \quad (73)$$
$$TA_{03} = TB_3 \cdot RotZ(\alpha_3) \cdot Transl(ServoArm, 0, 0) \quad (74)$$
$$TA_{04} = TB_4 \cdot RotZ(\alpha_4) \cdot Transl(-ServoArm, 0, 0) \quad (75)$$
$$TA_{05} = TB_5 \cdot RotZ(\alpha_5) \cdot Transl(ServoArm, 0, 0) \quad (76)$$
$$TA_{06} = TB_6 \cdot RotZ(\alpha_6) \cdot Transl(-ServoArm, 0, 0) \quad (77)$$

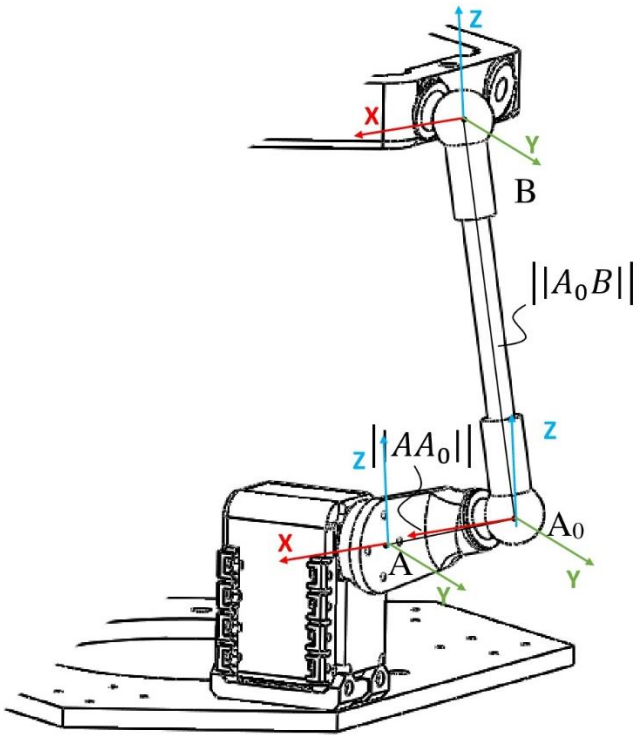


Figure 51 Reference in AA_0B Mechanism

Then we need to express the A_0 point at the same coordinates as the platform references.

$$TA_{pi} = TP_i^{-1} \cdot TA_{0i} \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} xa_i \\ ya_i \\ za_i \\ 1 \end{pmatrix} \tag{78}$$

Then we can compute the new vector by using the values from equation (8)

$$\vec{e}_i = \begin{pmatrix} xa_i \\ ya_i \\ za_i \end{pmatrix} - \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \tag{79}$$

5 SOFTWARE DESIGN

Once the electronics has been developed and tested and the mathematics needed to conclude the application has been developed the final step is to write all the code that will implement all the proper calculus.

The codes implemented in this project will run in different platforms, as the project includes a PC application and some microcontroller boards, also a graphic user interface was needed at the end of the project in order to interact with the system and visualize the obtained results.

Therefore, the codes developed in this section go from the basic low-level C++ programming of a microcontroller up to a more complex high-level object-oriented programming of the GUI.

5.1 FIRMWARE

As explained in Chapter 2.3.2 the manufacturer of the OpenCM 904 board provides some libraries to program the microcontroller as a master of the Dynamixel bus. Nevertheless, in this application we need to program it to behave as a slave of the bus, therefore a new library needs to be developed.

This library is going to be based around the library provided by the manufacturer as it offers some interesting functionalities, one is that the ROBOTIS library manages the hardware responsible of the half duplex serial communication, another one is that the ROBOTIS library creates a virtual serial port in Pins which do not have the built in hardware required, this is very useful as it will reduce the amount of low level coding.

These functionalities are available through the following functions:

- available(): returns true indicating if any data is available at the asynchronous serial bus to read.
- readRaw(): returns the byte read at the serial bus.
- writeRaw(): sends a byte through the serial bus.

The library developed will consist of a bunch of functions that will take care of:

- Loading the values from the EEPROM memory.
- Decoding the instructions sent from the master.
- Executing the possible orders.
- Returning the status packet.
- Reading the analogic signal from the conditioning circuitry.
- Controlling the peripherals through the i2c bus.

Register table

The first step in order to create a Dynamixel slave is to define the position of the registers in the EEPROM and the RAM table. The position of this registers is set in order to make them coincide with the ones that the actuators AX-12 have in common. Therefore, the same high-level functions may be used to access to the same register.
Then the defined register table for the board is:

Area	Address (Hexadecimal)	Name	Description
EEPROM	0(0x00)	Model Number(L)	Lowest byte of model number
	1(0x01)	Model Number(H)	Highest byte of model number
	2(0x02)	Version of Firmware	Information on the version of firmware
	3(0x03)	ID	ID of Dynamixel
	4(0x04)	Dxl Baud Rate	Baud Rate of Dynamixel
	5(0x05)	Return Delay Time	Return Delay Time
	16(0x10)	Status Return Level	Status Return Level
	17(0x11)	Baud Rate	Baud Rate of USB Port
	18(0x12)	Offset Error(L)	Lowest byte of offset error
	19(0x13)	Offset Error(H)	Highest byte of offset error
	20(0x14)	Gain Error(L)	Lowest byte of gain error
	21(0x15)	Gain Error(H)	Highest byte of gain error
RAM	24(0x18)	Led	Indicator Led on/off
	25(0x19)	Gauge(L)	Lowest byte of gauge signal
	26(0x1A)	Gauge(H)	Highest byte of gauge signal
	27(0x1B)	Voltage Supply(L)	Lowest byte of load cell supply voltage
	28(0x1C)	Voltage Supply(H)	Highest byte of load cell supply voltage
	29(0x1D)	Force(L)	Lowest byte of the computed force in mN[14]
	30(0x1E)	Force(H)	Highest byte of the computed force in mN
	31(0x1F)	Start Offset Calibration	Bit to start the offset calibration
	32 (0x20)	Calibration Offset done	Flag offset calibration finished
	33(0x21)	Calibrate Gain	Bit to calibrate the gain

Table 5 Conditioning Board Register Table

Finite-state machine

Once the Register Table has been defined the next step is to set the work flow of the code. As the library provided by the manufacturer only allows us to read the data byte by byte instead of the whole packet, the only way to reconstruct the original message is to create a finite-state machine which goes through the whole packet, decoding each data byte. The Block diagram of this machine is shown in the Figure 52.

[14] mN: miliNewotns

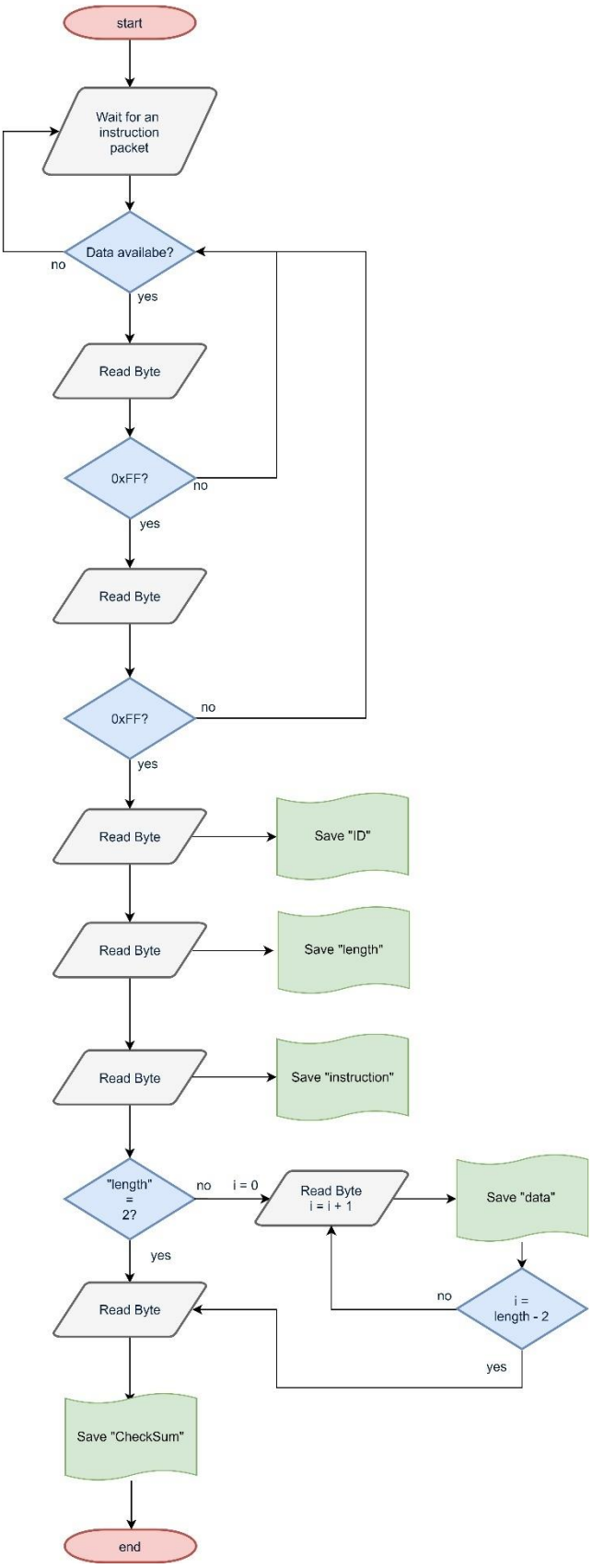


Figure 52 Block Diagram for Packet Decoding

At the moment the library is initialized, it will load the values from the EEPROM, then its design to set the GetMessage() function in the loop of the Arduino code, as it will wait until some instruction packet is received through the half-duplex serial port, then it will decode the message, check if the ID of the packet corresponds to the board and execute the proper order.

The code can be explained as a finite-state machine with the following diagram:

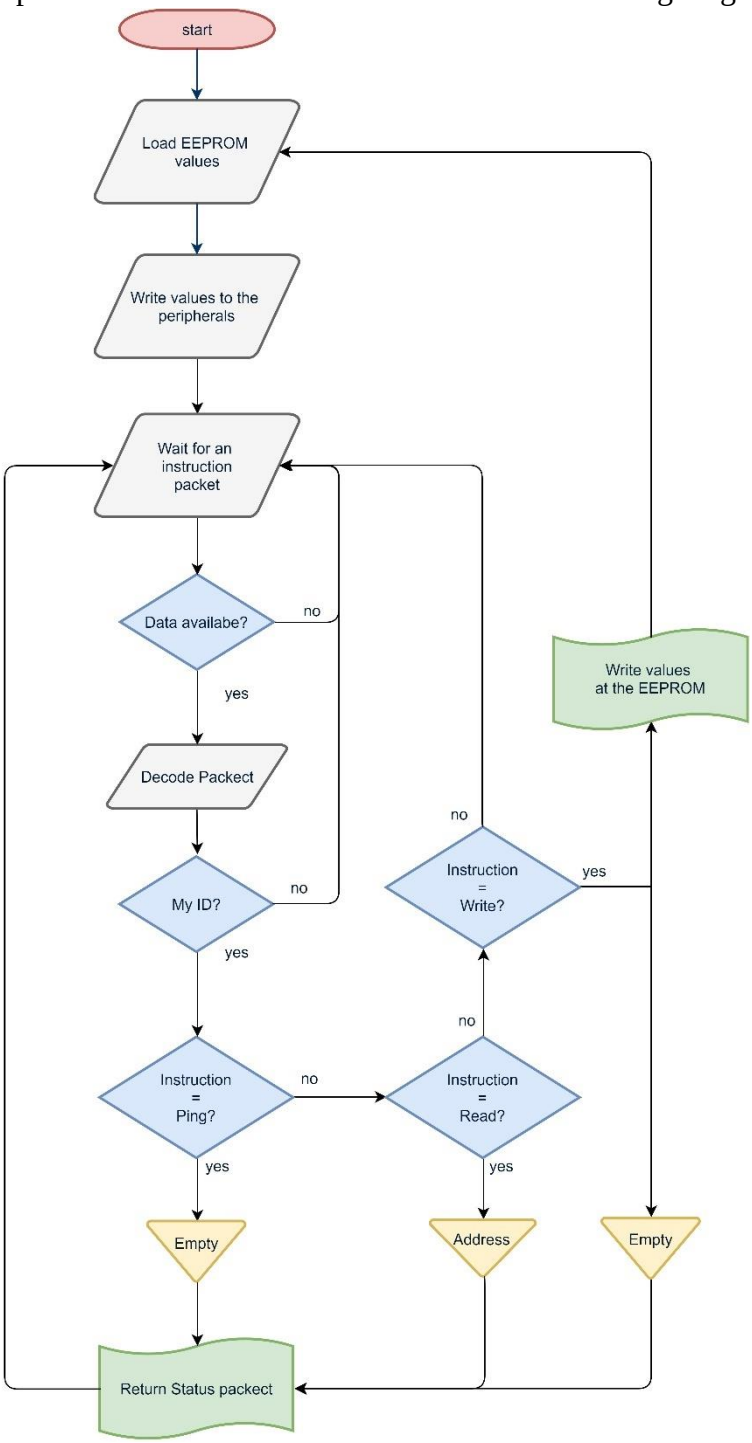


Figure 53 Firmware Block Diagram

At the same time the code needs to be configured to execute a timer event interrupt in order to execute the GetReadings() function. This function will read the ADC values at a fixed sample rate in order to then compute the force and save the values in the RAM register.

```
#include <SlaveCM904.h>
#include <SlowSoftI2CMaster.h>
#include <Wire.h>
#define myID 13

SlaveCM904 CM;
HardwareTimer Timer(1);
void setup()
{
  CM.Setup();
  CM.Begin();
  //CM.Set(MODEL_CM_904L, MODEL_CM_904H, FIRMWARE_S18, myID, DXL_BAUD_RATE_1Mbps, 10, 1, BAUD_9600, 0x0E,0x66,0,0);
  // Pause the timer while we're configuring it
  Timer.pause();
  // Set up period
  Timer.setPeriod(1000); //in microseconds
  // Set up an interrupt on channel 1
  Timer.attachInterrupt(handler_led);
  // Refresh the timer's count, prescale, and overflow
  Timer.refresh();
  // Start the timer counting
  Timer.resume();
}

void loop()
{
  CM.GetMessage();
}

void handler_led(void) {
  CM.GetReadings();
}
```

Figure 54 Timer interrupt configuration code

Finally, the development of the function Set() was done, the purpose of this function is to write in the EEPROM the essential values for the code to work when the chip is new and empty. Otherwise it will be impossible to run the code using an empty EEPROM.

The whole library and the Arduino example code can be found in Appendix J.

5.2 DRIVER

Once the firmware responsible of controlling the microcontroller has been developed and more important, the register table is defined, we can proceed to create the driver.

The main purpose of creating a driver is to ease the task of communicating to the device, i.e. we could use the IRI Dynamixel library to either read or write any value to the registers of the microcontroller board, but it will require us to remember which is the address that corresponds to each register, the length of the data, etc.

Therefore, the driver will be a set of functions that will allow us to access to any register and exploit all the capabilities of the microcontroller boards. Those functions are:

- **GetForce():** Returns the value of the force read (in mN) casted to an integer.
- **GetError():** Returns the gain error.
- **ReadVoltage():** Returns the bit value of the reading of the ADC.
- **ReadSupply():** Returns the voltage of the supply of the Load Cell.
- **CalibrateOffset():** Starts the process of calibrating the offset, returns true when finished.
- **CalibrateGain():** Calibrates the gain of the circuit.
- **CheckLed():** Turns On, Off or Toggles the Led.
- **SetFilter():** Sets a threshold value to return the force only if is bigger.

The whole driver and an example code can be found in Appendix K.

5.3 APPLICATION

Finally, the last part of the software is the application, this is the main code that will combine the mathematical work done in Chapter 4, the implemented driver mentioned in Section 5.2 and all the different libraries explained in Chapter 2.3 either the ones developed at the IRI or the external ones.

The purpose of this application is to ask periodically to each one of the Conditioning Boards for the reading of the force that is being applied through the legs of the platform, then the application will compute the information in order to recalculate the original point of application of the force by following the steps explained in Chapter 4.2, the matrix calculation has been implemented using the Eigen library.

Once the desired position has been calculated, the required position of the actuators needed to obtain it, is computed by following the steps described in Chapter 4.1, as well as before, the matrices calculations were implemented using Eigen.

In order to repeat the process of requesting the forces to the Conditioning Boards and sending the goal positions to the actuators periodically, at a well-known frequency, a thread code was implemented in this application.

Multithreading is the ability of a code to execute multiple processes or threads concurrently. These threads share the process's resources but are able to execute independently. Those multiple threads are capable to communicate between them easily and fast by sharing variables. In this case, the class “Thread” from the **iriutils** library was used to create the thread. Nevertheless, other options were tested, as the C++11 functionalities[15].

5.4 GRAPHICS USER INTERFACE

Finally, the last shield of software is the Graphic User Interface, this shield is not mandatory for the correct work of the project. Nevertheless, it highly facilities the functionalities of the code making it much more visual and intuitive for a user to use.

The GUI is splited in three sections, each one will be enabled depending on which devices are connected:

- **Manual Actuation:** it will be enabled to use when only the actuators are connected. This section allows the user to move manually the platform through the interaction of 6 sliders, three for the translations in the three axis, and three for the rotations in the three axes.
- **Reading Forces:** it will be enabled either when only the conditioning boards are connected or when both conditioning boards and actuators are connected. This section allows to see the reading force of each Conditioning Board, the computed position and the filtered one, start the offset calibration, select the sample frequency and much more functions.
- **Actuation:** it will be enabled when only both conditioning boards and actuators are connected. This section allows to define the resistance in both axis, the movement speed of the actuators and enable the main actuation.

[15] C++ 11 Thread class: https://en.cppreference.com/w/cpp/thread/thread

The connection of the devices can be selected at the top of the interface.

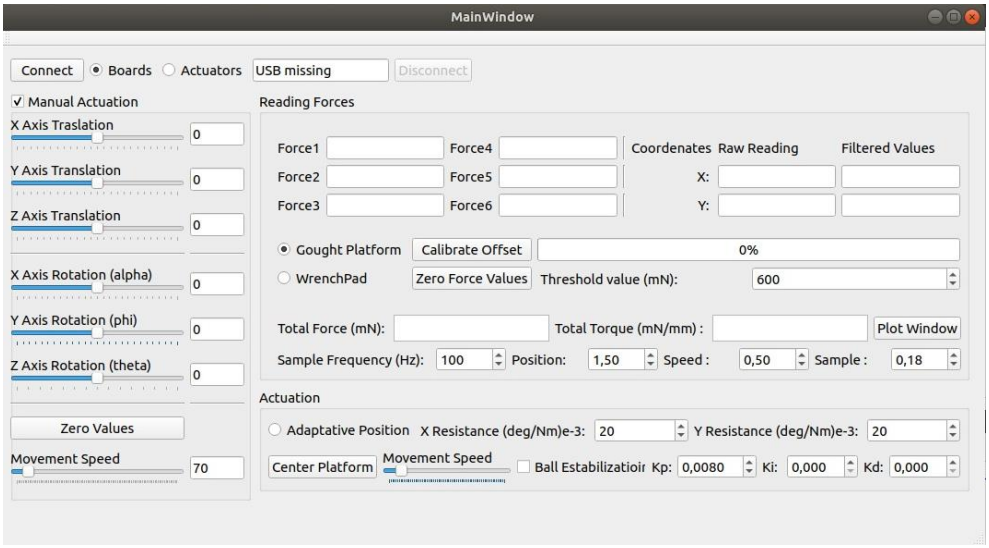


Figure 55 Main Window of the GUI

When the Reading Forces section is enabled, the Plot Window button can be clicked, if done, the application will open a new dialog to plot the position coordinates of the force, this dialog shows a plot axis of the size of the platform, and some check buttons to choose the kind of plot desired.

Figure 56 Plot Dialog of the GUI

From Figure 56 we can see the Dialog used to Plot the data in real time, this window allows to select the mode of plot, it can plot just the position in real time or draw the line of the path the force applies, it also allows us to visualize either the raw data or the filtered one.

The code used to developed this GUI can be found in Appendix M.

6 RESULTS

Finally, in this section the results obtained from the work done in the project are reviewed. First of all we are going to go through the results obtained in the process of getting the position of the force, that includes the communication between the computer and the boards, the conditioning circuit (which results were reviewed in Chapter 3.4.1), and the mathematical background.

Once the force/torque sensor is reviewed we are going to focus in the application itself and how the system behaves.

6.1 FORCE/TORQUE COMPARISON

In order to check how the system performs objectively, we need a controlled environment, therefore, we decided to test the system in the same platform where the article [1] was developed. To do it we need to reconfigure the gain of the system, as the load cells installed in that platform are much more sensitive.

Once the boards have been reconfigured the code found in Appendix L has been used to save the values of the readings of the Boards into a .txt file for a period of time. Then a Matlab Script Appendix H.1 has been written to open the .txt file and show the following results:

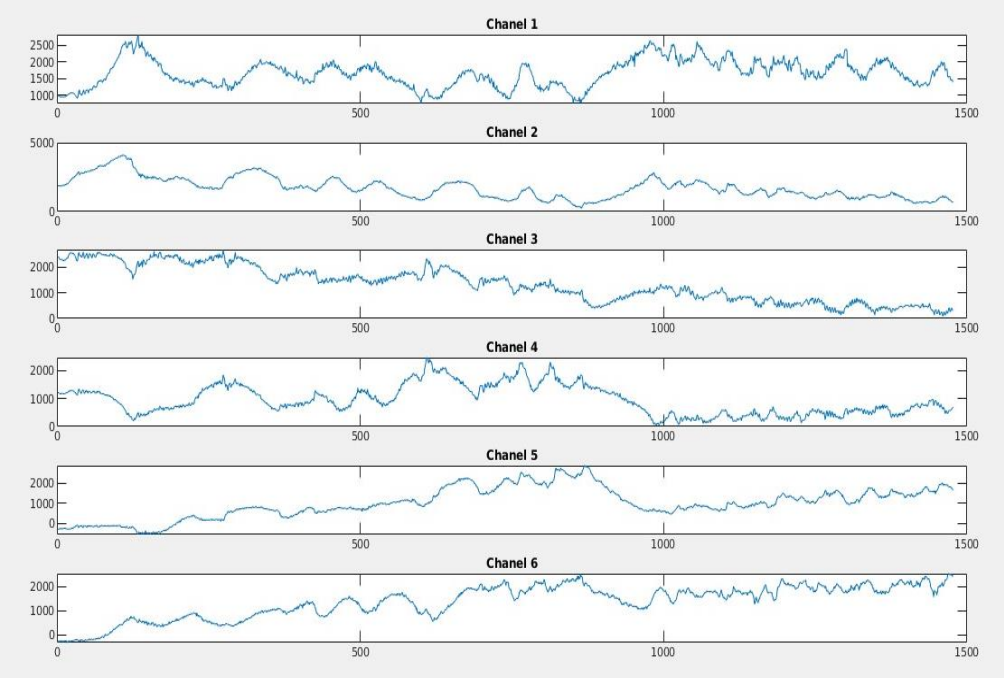


Figure 57 Reading of 6 Forces

As we can see from Figure 57 the readings respond quite well to the path of the force that was exercised at the platform. On the other hand, we can see that the readings are quite noisy.

To check how the noise, we have found at the readings, affects at the process of calculating the point of application another Matlab script was written (Appendix H.2) in order to test the calculations done in Chapter 4.2, from there we can see the X/Y plot of the recomputed position:

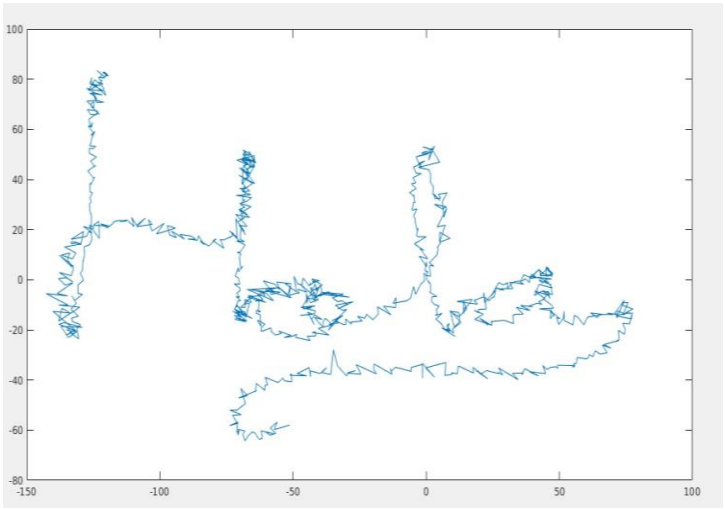


Figure 58 XY Plot of noisy values

As we can see from Figure 58 the resulting position is quite awful, fact that was totally unexpected. This made us think that there was something wrong in our system that result in this extremely noise position estimation.

The first hypothesis was that the delay time between samples was causing this distortion, as the register in each board gets updated periodically through a timer interrupt and the application only can ask for a reading at a time, the samples of the forces does not correspond to the same instant. Therefore, it is possible to think that this delay between samples can cause some kind of distortion.

In order to test this hypothesis, the firmware was temporally modified in order to make the boards update their register only when a global ID sends a message, this way, the readings will be updated at the same time. Allowing to then transfer them one by one.

Sadly though, this test did not show any changes making the shown position as distorted as before.

The second hypothesis was that the small noise found at the readings translates into a high uncertainty during the calculus of the position. To test if this was the case the Matlab scrip from Appendix H.1 was used to show the DFT of the signal:

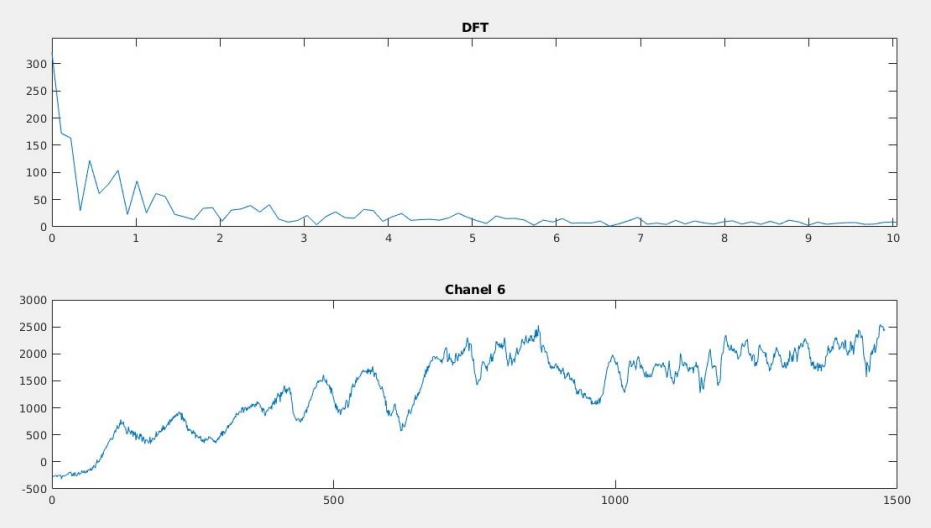


Figure 59 DFT of channel 6

As we can see from Figure 59 the Fourier transform shows a progressive decrease of the amplitude of its harmonics without showing any clear spike, this fact makes much more difficult to identify the threshold point where the valuable information differentiates from the noise. By following a process of trial and error we found that about 5Hz was the limit of the relevant information.

Then, a digital low-pass filter of 8th order at 5Hz was developed to filter the data.

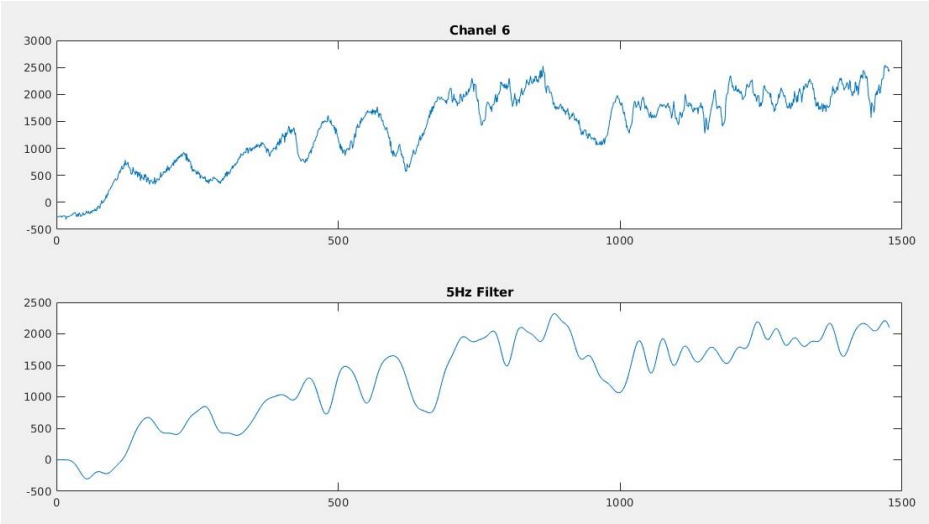


Figure 60 Original data vs. Filtered data

Figure 60 shows the resulting data of channel 6 after being filtered, it is clear that the filters highly reduce the noise resulting in an extremely clear path of the force. Then the second Matlab script (Appendix H.2) was modified to show the X/Y plot after filtering the readings.

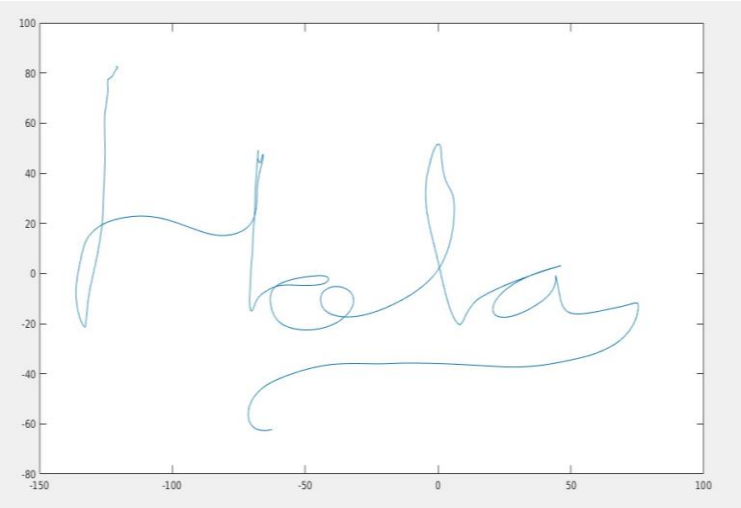


Figure 61 X/Y Plot of filtered readings

Finally, when showing the X/Y plot we can see how all the noise in the position was removed, fact that made clear that the noise found at the readings was responsible of the uncertainty at the position even being a small one.

From the image in Figure. 38 we could think that applying a digital low-pass filter to the readings could be a definitive solution to the problem. Nevertheless, event though the low-pass filter results in an extremely round and perfect shape it is not. The low pass filter distorts the real information when removing its noise, which results in a position that does not correspond to the real reding, it is a distorted one. To remove the noise without losing the real information about the position we decided to implement a Kalman filter.

6.1.1 Kalman Filter

The Kalman filter is a recursive state estimator for non-stationary processes, thanks to which optimal estimates can be obtained from noisy measurements. It was developed in the early sixties and was quickly used in many areas of knowledge, from orbiting navigation at the Apollo project to image processing or radar guidance.

In our case, the Kalman filter will allow tracking the point of application of the outer force on the platform by filtering the noise that exists in the measure or the displacement of the force on the platform.

A linear system in a discrete time will be used as a model in which the state vector will be formed by the position and the speed on the platform.

First of all the equations that define a Kalman filter need to be explained, There are two types of equations for the Kalman filter. The first are the **prediction** equations. These equations predict what the current state is based on the previous state. The second set of equations known as the **update** equations look at the input, how much you trust each input, and how much you trust your overall state estimate. This filter works by predicting the current state using the prediction equations followed by checking how good of a job it did predicting by using the update equations. This process is repeated continuously to update the current state.

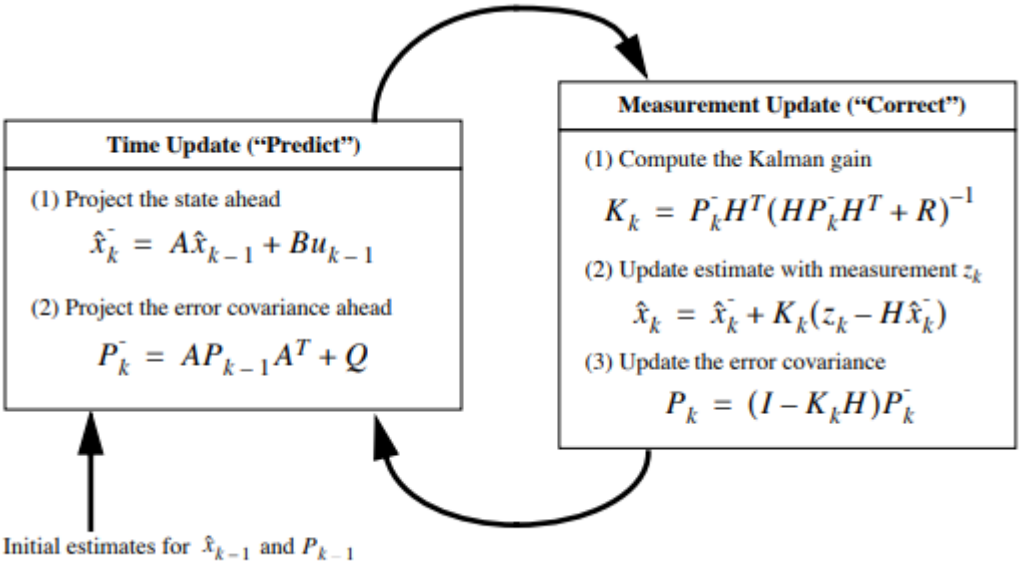


Figure 62 Kalman filter equations

This equations contain a lot of equations, so we will now clarify what each variable is. First of all is mandatory to explain that the $K-1$ reefers to the previous state and *K* to the new state.

- *X*: is the state, i.e. the things we are trying to control. Therefore, in our case it will be a $\mathbb{R}^4$ vector: $x = (pos_X, pos_Y, v_x, v_Y)^\top$.
- *A*: is the matrix that relates the state at the previous time step $K-1$ to the state at the current step *K*. Therefore,

$$A = \begin{pmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (80)$$

- *u*: is the action used to control the system. In our case there is none, therefore, *u=0.*
- *B*: is the matrix that relates the control input *u* to the state *x*. In our case also 0.
- *P*: is the error covariance. These numbers represent how confident the filter is with the solution. The best way to do this is to initialize it as a diagonal matrix when the filter runs it will become populated.
- *Q*: Is the covariance of the process (i.e. action) noise. It is formed similar to the above except that it is a matrix that you determine and does not get updated by the filter. This matrix tells the Kalman filter that the trajectory of the point on the platform does not necessarily have to follow the extraordinarily simple model represented by *A* but that there may be changes in the direction of movement.

$$Q = \begin{pmatrix} \sigma_{pos} & 0 & 0 & 0 \\ 0 & \sigma_{pos} & 0 & 0 \\ 0 & 0 & \sigma_v & 0 \\ 0 & 0 & 0 & \sigma_v \end{pmatrix} \quad (81)$$

where σ_{pos}, σ_v refer to the variance in the real position of the point.

- *K*: This matrix gets updates as part of the measurement update stage.
- *H*: Is a model of the sensors, it will be considered that the sensor directly gives the measure of the position on the platform. So,

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{82}$$

- *R*: This matrix is similar to *Q* except this defines the confidence in each of the sensors.

$$R = \begin{pmatrix} \sigma_{sample} & 0 \\ 0 & \sigma_{sample} \end{pmatrix} \tag{83}$$

where σ_{sample} is the uncertainty in the measure.

- *z*: These are the measurements that are returned from the sensors at each location.

Once the mathematical work of the Kalman filter has been done, a third Matlab script (Appendix H.3) was written in order to verify its functionalities.

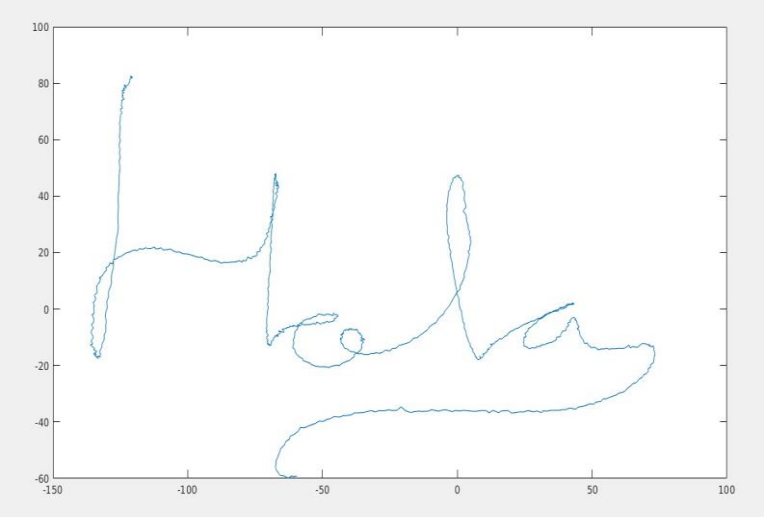


Figure 63 Kalman filtered position

From Figure 63 we can clearly see that the Kalman filter estimates perfectly the position of the force, some noise still remains present at the plot but this time the estimated position conserves the actual position.

6.1.2 Redefinition of work range

As the previous section shown, the noise present in the readings of the forces is too significant, as it creates a high uncertainty in the calculation of the position. Analysing the output of one of the boards we have found that the noise at the output corresponds to the last 2 bits of the DAC.

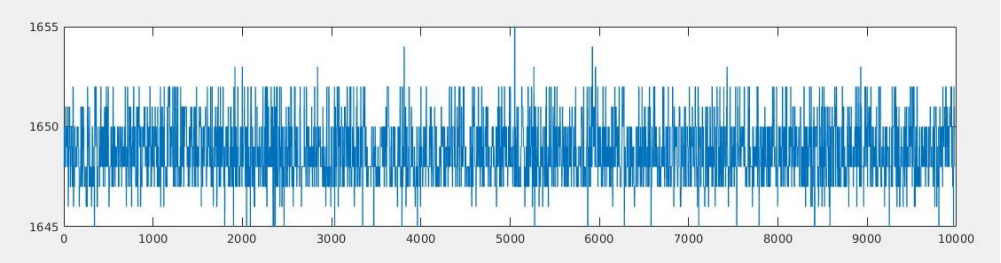


Figure 64 Noise at the reading of the force

Now that the circuity is already built we can't modify its design to reduce the noise, but it is possible to redefine its work range.

In the final design of the circuit two assumptions were made:

- The load each cell will have to deal is of ±2Kg.
- The useful information will be found between 0 and 100Hz.

Modifying the first one will result in reducing the amount of weight each bit of noise will be carrying, modifying the second one will reduce the quantity of undesired harmonics of noise.

Now that we have the real system build we can test how accurate dose assumptions were, then we can redefine the gain of the circuit and the cutting frequency of the filters to match its necessities. To do it to tests were performed:

- **Static Test:** In order to find the maximum force the load cells will deal with, a user wrote in the platform some words in different positions and applying different forces to it.
- **Dynamic Test:** In order to find the maximum frequency where the useful signal will be placed, a user wrote in the platform some words at different sizes and speeds.

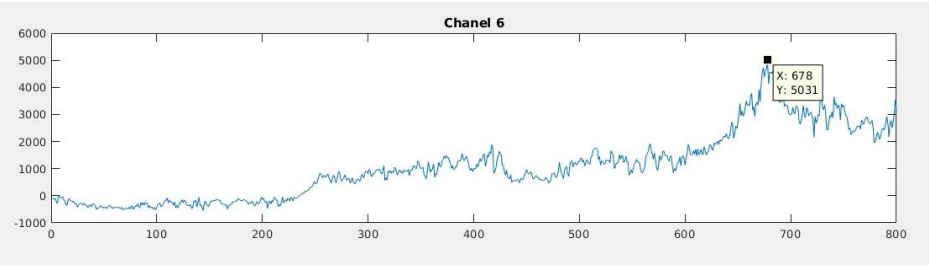


Figure 65 Static test results

The static test shown a maximum force of 5032mN, which multiplied by a security factor of 1.1 results in a value of ±550g in each load cell.

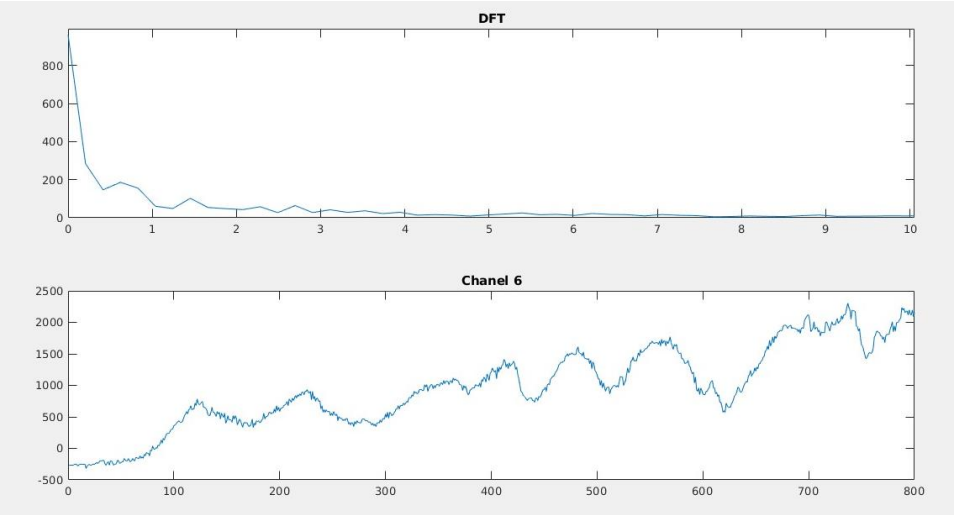


Figure 66 Dynamic tests results

The resulting DFT of the dynamic test was not conclusive enough as well as in section 6.1, therefore the information obtained before and some trial and error with different digital filters conclude in a cutting frequency of 5Hz, which multiplied by a security factor of 1.5 results in a value of 7.5Hz.

Once the new values have been found we proceed to calculate the proper values for the new components required to match those necessities.

Results of the new work range

Once the necessary hardware has been changed the circuitry has been tested with the new gain and the new filters. The result of this redefinition of the work range has shown the expected result, the amount of noise is exactly the same but in this case it is much less significant, so the position estimation looks much more accurate.

6.2 ACTUATION

Finally, once we have achieved the best-possible performance of the force/torque sensor we can proceed to verify de behaviour of the actuation.

The only important thing to mention in this section is the balance that needs to be found between the speed of the actuators and the Sample Frequency. As this work does not consider the dynamics of this platform, each of the measures took by the boards needs to be done when no movement is applied to the platform, otherwise, the readings will correspond to the sum of the external force and forces involved in the movement of the platform. Therefore, the time between samples needs to be high enough for the actuators to achieve its goal position.

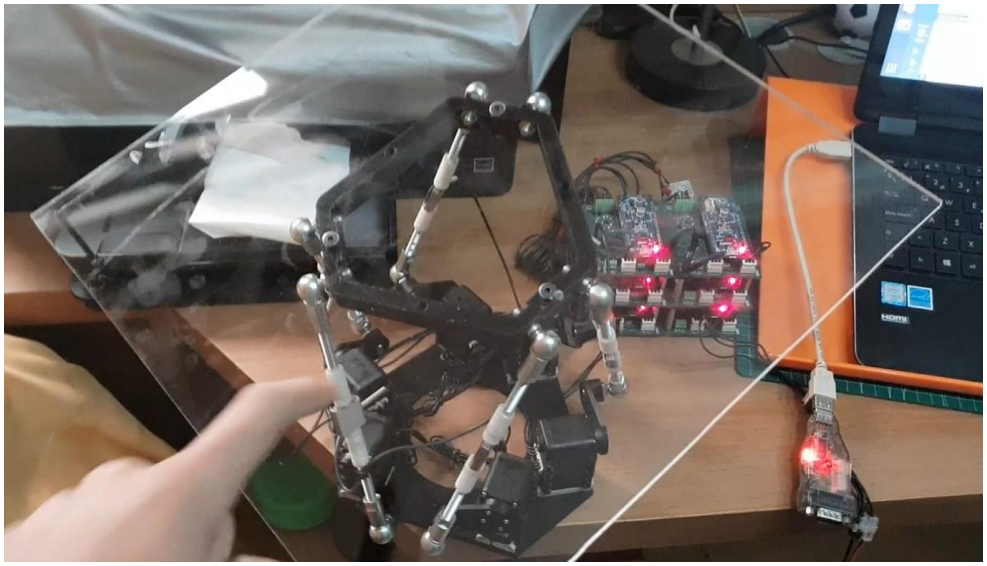


Figure 67 Platform Acting over a Force

As the final values can not be observed and analysed in this thesis, in the Conclusion chapter, the discussion of results is done quantitatively.

In order to show the system in action a video has been attached with this thesis.

7 CONCLUSIONS AND FUTURE WORK

A parallel robot of 6 degrees of freedom has been built with the preestablished specifications, as it has been able to successfully complete the control tasks for which it was designed. It is concluded therefore that it is powerful enough to provide a control such as the developed in this work and others of superior complexity. In order to be able to control this robot, the reverse kinematics has been solved. The study an analysation of the point of application has resulted in being very successful thanks to the application of a Kalman filter, which has proven to be a very accurate and useful estimator.

Throughout the development of this project many complications have emerged. Most of them related with the design and build of the PCBs, more than one design iterations of either the circuit itself or the PCB has been needed. Some big problems like a wrong design of the PCB or having to deal with the delivery times of the providers have highly delayed the project. The limited time and the choice of the devices, still not the most appropriate, have made us take the most pragmatic choice to have a system working at the end of the project. Here it is an enumeration of the possible improvements which should be implemented to have a more robust system if someone else wants to continue with this researching project.

- The first clear improvement of this project comes from the Conditioning Boards. The fact that 6 boards are used to conditionate the signal of the 6 load cells has proven to work really well. Is a system very reliable as it offers the possibility to reuse this boards for another project were the number of forces to measure might be higher or lower. Nevertheless, if someone would like to keep working on this specific project it should combine the 6 circuits in a single PCB with only one microcontroller, that way not only the total cost will be reduced but also the sample frequency of the system cold be increased, as it will allow to read the 6 forces by making only one request.
- In chapter 6.1 the presence of noise in the readings has shown how relevant it really is. One way to reduce it is to modify the circuitry in order to substitute the actual passive filtering method for an active filter system of high order. This will increase the chances of removing the undesired noise that causes the uncertainties when calculating the position of the force.
- Another option is to realize a deeper study of the noise propagation. One study of this characteristics may increase the chances of finding a clear source of the noise, if so, the proper circuitry to remove it may be applied.

- One hypothesis postulated during this project is that the source of the noise might come from using the same voltage supply (inside the circuit) for the analogic and digital circuitry. Even though the power planes were separated to avoid the coupling of digital noise some of them might still be present, one possible solution is to increase the isolation by using an inductor instead of a high impedance zone, another option will be to separate the voltage supplies into one for each part.
- In chapter 6.1 we also saw how relevant a bit of information translates into the position estimation, this proved that the number of bits chose for the DAC was to low and if a more precise estimation needs to be done the number of bits should be increased.
- The work of this project has successfully proven that only a kinematic study is needed when dealing with parallel manipulators of this weight and size. Nevertheless, this is only possible under certain conditions (tweaking the sample frequency and the speed of the actuators), in order to create a much more reliable system capable of working in all conditions the proper dynamic study should be done.
- The Kalman filter has end up being one of the key components responsible of the success of this project, one of its interesting features is that it allows us to tell him the amount of uncertainty we have in each reading, as proven in [1] the uncertainty found in the readings of each force does not translate lineally to the position but ellipsoidly. In this project a lineal dispersion was assumed, but if the proper study of the uncertainty dispersion is done it will allow us to tell to the Kalman filter the amount of uncertainty per each position sample, remaining in a much more accurate filter.
- Finally, the platform is currently reacts only to the perpendicular forces applied at the top plate in consequence to the distance from the centre point, this leads the platform to be a compliant mechanism of only two axis. Even though this already is a quite complex system it will be much more interesting if it could move consequently to any force applied at any axis. This opens a new problem as some work has already been done in finding the point of application over the plate [1] or converting the platform in a six axis sensor [2] but not combining both.

8 BIBLIOGRAPHY

[1] Frigola Alcalde, R., Ros Giralt, L., Roure Fernández, F., & Thomas, F. (2008). A wrench-sensitive touch pad based on a parallel structure. In 2008 IEEE International Conference on Robotics and Automation (pp. 3449-3459).

[2] Ruiz, M. R. (2017). Design and analysis of a Stewart-platform-based six-axis load cell (Doctoral dissertation, Massachusetts Institute of Technology).

[3] Ranganath, R., Nair, P. S., Mruthyunjaya, T. S., & Ghosal, A. (2004). A force–torque sensor based on a Stewart Platform in a near-singular configuration. *Mechanism and machine theory*, *39*(9), 971-998.

[4] Staicu, S. (2011). Dynamics of the 6-6 Stewart parallel manipulator. *Robotics and Computer-Integrated Manufacturing*, *27*(1), 212-220.

[5] Szufnarowski, F. (2013). Stewart platform with fixed rotary actuators: a low cost design study. *Advances in Medical Robotics*.

[7] Refaat, S., Hervé, J. M., Nahavandi, S., & Trinh, H. (2007). Two-mode overconstrained three-DOFs rotational-translational linear-motor-based parallel-kinematics mechanism for machine tool applications. Robotica, 25(4), 461-466.

[8] Machiani, J. H., Masouleh, M. T., Kalhor, A., Tabrizi, M. G., & Sanie, F. (2014, October). Control of a pneumatically actuated 6-DOF Gough-Stewart platform. In *Robotics and Mechatronics (ICRoM), 2014 Second RSI/ISM International Conference on*(pp. 166-171). IEEE.

[9] Ghosal, A. Six component force-torque sensors using Gough-Stewart platform manipulators.

[10] Szufnarowski, F. (2013). Stewart platform with fixed rotary actuators: a low cost design study. Advances in Medical Robotics.

[11] J.-P. Mertlet. (2006) Parallel robots 2nd Edition Ed. Springer pp. 131-132

[12] Analog Devices; Understanding and Designing Differential Filters for Communications Systems: http://www.analog.com/en/technical-articles/understanding-and-designing-differential-filters-for-communications-systems.html

[13] RT; LOAD CELL CABLING
http://www.marco-iw.nl/file/1355354080.2319WAfreS/2%20Loadcell%20cabling.pdf

[14] Jeffrey R. Riskin, Analog Devices; A User's Guide to IC Instrumentation Amplifiers: http://www.analog.com/media/en/technical-documentation/application-notes/AN-244.pdf

[15] Kitchin, C. (2007). Avoid common problems when designing amplifier circuits. *Analog Dialogue*, *41*(08), 1-4.

[16] Nicolas Aupetit, ST; How to filter the input of a high-side current sensing:
https://www.st.com/content/ccc/resource/technical/document/application_note/a3/84/7c/19/21/a8/40/f3/DM00086777.pdf/files/DM00086777.pdf/jcr:content/translations/en.DM00086777.pdf

[17] Kitchin, C., & Counts, L. (2005). The right way to use instrumentation amplifiers-Avoid common application problems when connecting real-world signals to instrumentation amplifiers. EDN, 50(19), 69-76.

[18] Teel, J. C. (2005). Understanding noise in linear regulators. Texas Instruments Analog Applicant.

[19] Mensink, A. (2008). Characterization and modeling of a dynamixel servo. *Trabajo Individual de Investigación en el Electrical Engineering Control Engineeringde la University of Twente*.

[20] Andrade-Cetto, J. (2002). The Kalman Filter.

[21] Bishop, G., & Welch, G. (2001). An introduction to the Kalman filter. *Proc of SIGGRAPH, Course*, *8*(27599-3175), 59.

[22] "ROBOTIS e-Manual for Dynamixel AX-12A/AX-12+." http://support.robotis.com/en/product/actuator/dynamixel/ax_series/dxl_ax_actuator.htm

[23] "ROBOTIS user's manual for Dynamixel." http://www.trossenrobotics.com/images/productdownloads/AX-12(English).pdf

[24] "Dynamixel AX-12A/AX-12+ Datasheet." http://www.pishrobot.com/files/products/datasheets/dynamixel_ax-12a.pdf

[25] "Utilities IRI library" http://wiki.iri.upc.edu/index.php/Utilities_library

[26] "Communications IRI library " http://wiki.iri.upc.edu/index.php/Communications_library

[27] "Dynamixel IRI library" http://wiki.iri.upc.edu/index.php/Dynamixel_library

[28] "Motor Control IRI library" http://wiki.iri.upc.edu/index.php/Motor_control_library

[29] "Cmake IRI info" http://wiki.iri.upc.edu/index.php/Cmake_info

Appendix A

Images and tables sources

Figure:	Source:
Figure 2	Frigola Alcalde, R., Ros Giralt, L., Roure Fernández, F., & Thomas, F. (2008). A wrench-sensitive touch pad based on a parallel structure. In 2008 IEEE International Conference on Robotics and Automation (pp. 3449-3459).
Figure 6	https://www.ram.ewi.utwente.nl/aigaion/attachments/single/1015
Figure 7	https://www.ram.ewi.utwente.nl/aigaion/attachments/single/1015
Figure 8	http://support.robotis.com/en/product/controller/opencm9.04.htm
Figure 9	https://www.robot-advance.com/EN/art-usb2dynamixel-pc-interface-to-bioloid-bus-945.htm
Figure 13	https://learn.sparkfun.com/tutorials/i2c
Figure 15	http://emanual.robotis.com/docs/en/software/arduino_ide/
Figure 16	https://es.wikipedia.org/wiki/Qt_(biblioteca)
Figure 18	http://journals.sagepub.com/doi/pdf/10.1177/1550147717722154
Figure 19	http://journals.sagepub.com/doi/pdf/10.1177/1550147717722154
Figure 21	https://www.allaboutcircuits.com/textbook/semiconductors/chpt-8/the-instrumentation-amplifier/
Figure 22	http://www.ti.com/lit/ds/symlink/tps735.pdf
Figure 23	http://ww1.microchip.com/downloads/en/DeviceDoc/20005318A.pdf
Figure 24	https://www.maximintegrated.com/en/app-notes/index.mvp/id/4602
Figure 31	http://www.ti.com/lit/ds/symlink/lm1117.pdf
Figure 35	http://www.circuitdomain.com/Creating%20Artwork/Creating%20Artwork.htm
Figure 36	http://www.mokopcb.com/moko-services/4-layer-pcb/
Figure 38	https://en.wikipedia.org/wiki/File:Stripline_stub_matching.svg
Figure 50	Frigola Alcalde, R., Ros Giralt, L., Roure Fernández, F., & Thomas, F. (2008). A wrench-sensitive touch pad based on a parallel structure. In 2008 IEEE International Conference on Robotics and Automation (pp. 3449-3459).
Figure 62	http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

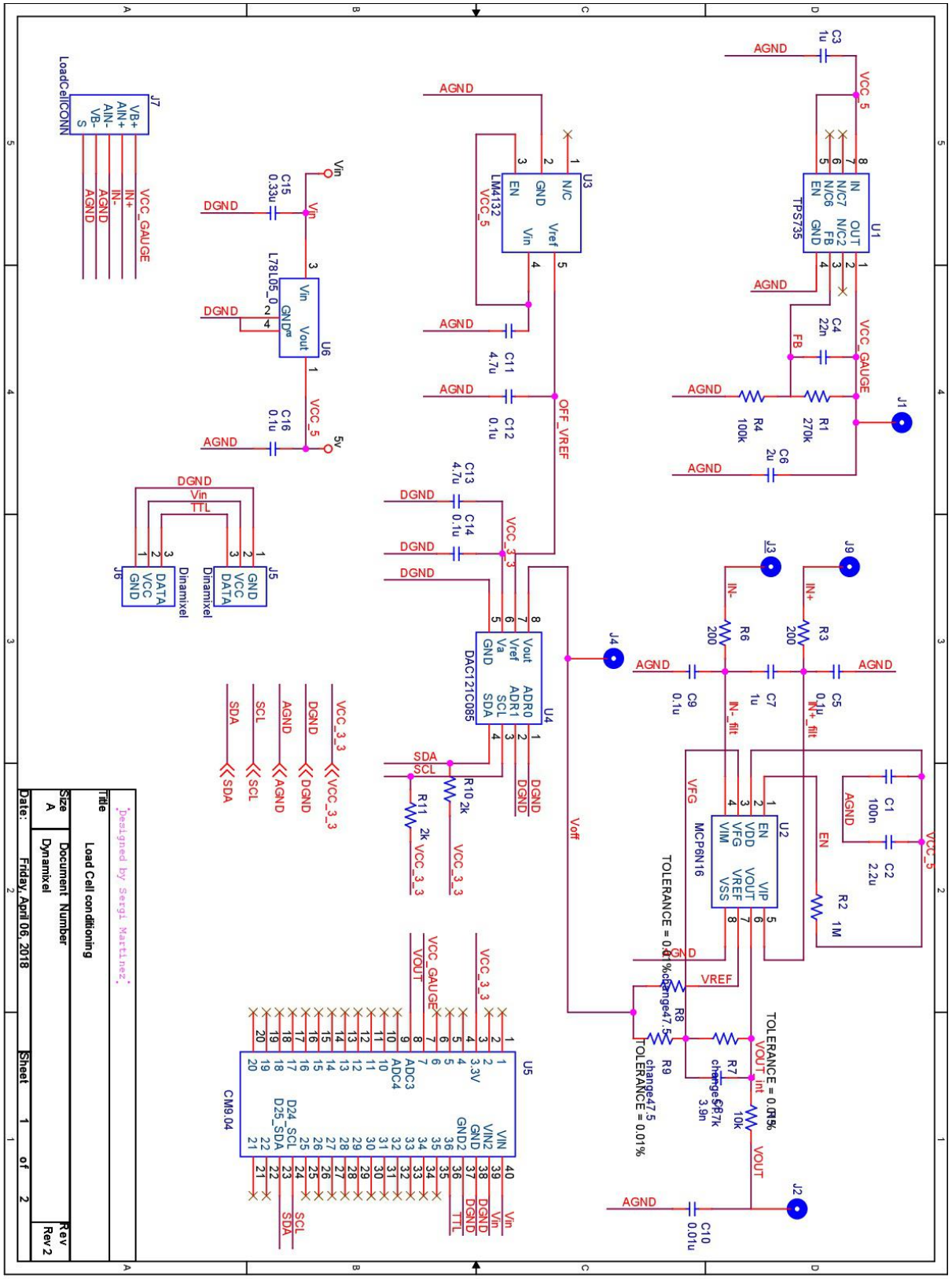
Appendix B

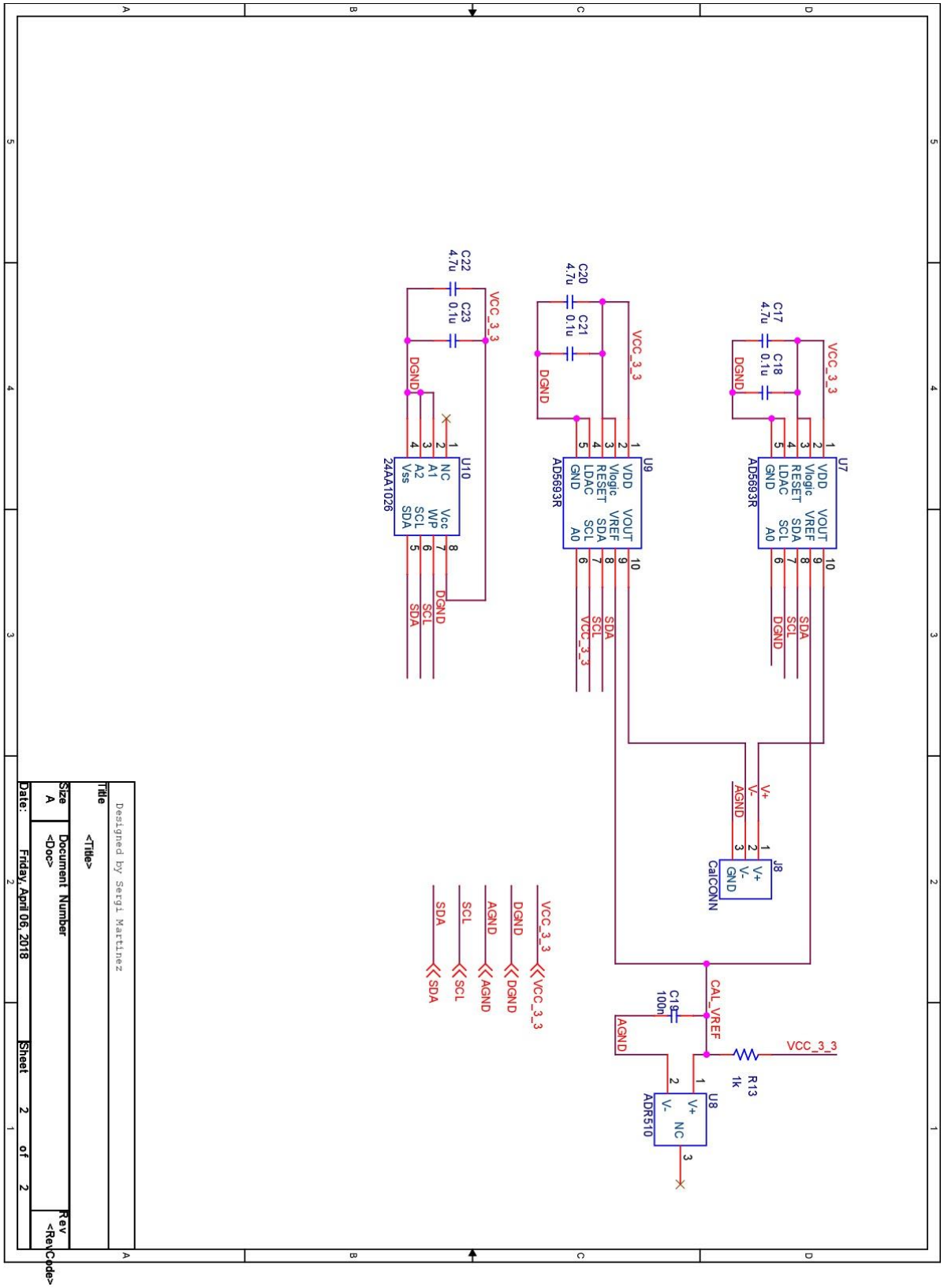
Circuit Bill of Materials

Item	Quantity	Reference	Part	RS Code	Price (unity)	Minimum amount	Total price
1	1	C2	2.2u	220-8010	0,077	50	3,85
2	2	C3,C7	1u	264-4450	0,062	25	1,55
3	1	C4	22n	798-4706	0,124	20	2,48
4	10	C1,C5,C9,C12,C14, C16,C18,C21,C23,C19	0.1u	648-0941	0,106	10	1,06
5	1	C6	2u	312-3156	0,165	10	1,65
6	1	C8	3.9n	148-238	0,39	25	9,75
7	1	C10	0.01u	264-4371	0,039	25	0,975
8	5	C11,C13,C17,C20,C22	4.7u	698-3579	0,052	10	0,52
9	1	C15	0.33u	802-9872	0,312	25	7,8
10	5	J1,J2,J3,J4,J9	TestCONN				0
11	2	J5,J6	Dinamixel	687-8124	0,103	10	1,03
12	1	J7	LoadCellCONN	790-1105	1,512	5	7,56
13	1	J8	CalCONN	790-1092	0,958	5	4,79
14	1	R1	270k	666-2711	0,466	5	2,33
15	1	R2	1M	807-5730	0,013	100	1,3
16	2	R3,R6	200	828-0620	0,36	5	1,8
17	1	R4	100k	566-765	0,274	5	1,37
18	1	R5	10k	721-7820	0,224	5	1,12
19	1	R7	51k	662-1379	0,864	5	4,32
20	2	R8,R9	47.5	614-5187	0,756	5	3,78
21	2	R10,R11	2k	666-2547	0,158	5	0,79
22	1	R13	1k	565-964	0,274	5	1,37
23	1	U1	TPS735	817-5389	1,357	10	13,57
24	1	U2	MCP6N16	829-0560	2,114	5	10,57
25	1	U3	LM4132	407-206	3,12	1	3,12
26	1	U4	DAC121C085	517-123	2,4	1	2,4
27	1	U5	pins CM9.04	267-7387	5,61	5	28,05
28	1	U6	L78L05_0	714-0679	0,314	10	3,14
29	2	U7,U9	AD5693R	831-9107	7,31	1	14,62
30	1	U8	ADR510	819-7070	3,284	5	16,42
31	1	U10	24AA1026	823-0650	3,175	2	6,35

Appendix C

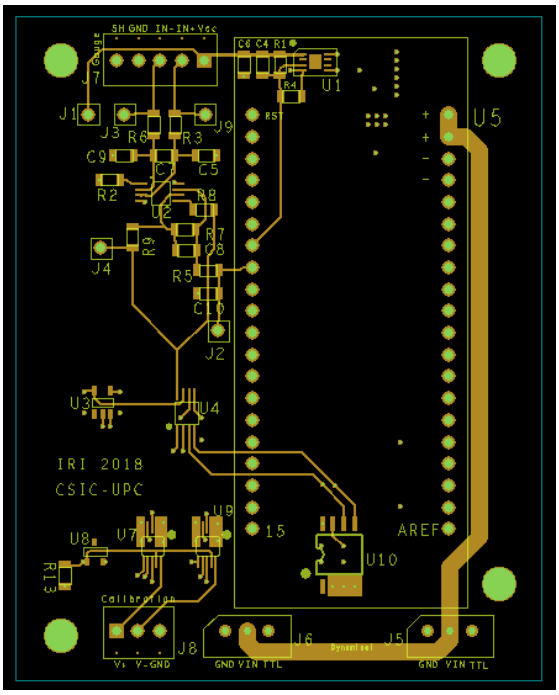
Circuit Schematic



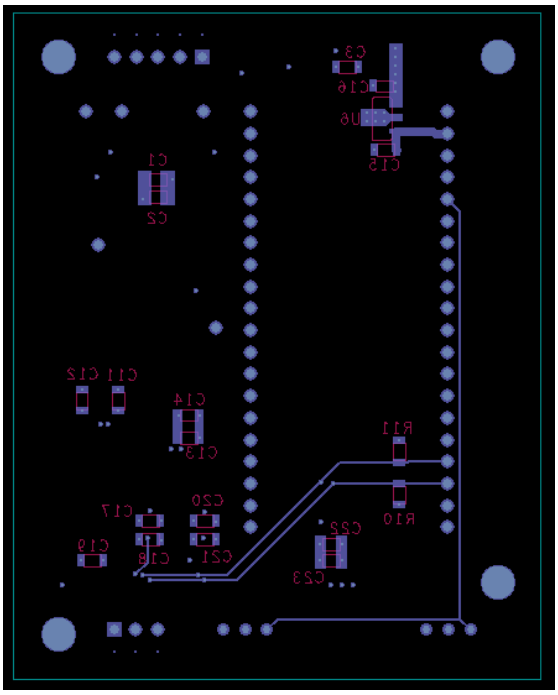

U7
AD5693R
U9
AD5693R
U10
24AA1026
U8
ADR510
J8
CalCONN
VCC_3_3
DGND
AGND
SDA
SCL
CAL_VREF
R13
1k
C17
4.7u
C18
0.1u
C20
4.7u
C21
0.1u
C22
4.7u
C23
0.1u
C19
100n
Designed by Sergi Martinez
Title
<Title>
Size
A
Document Number
<Doc>
Rev
<RevCode>
Date:
Friday, April 06, 2018
Sheet
2
of
2

Appendix D

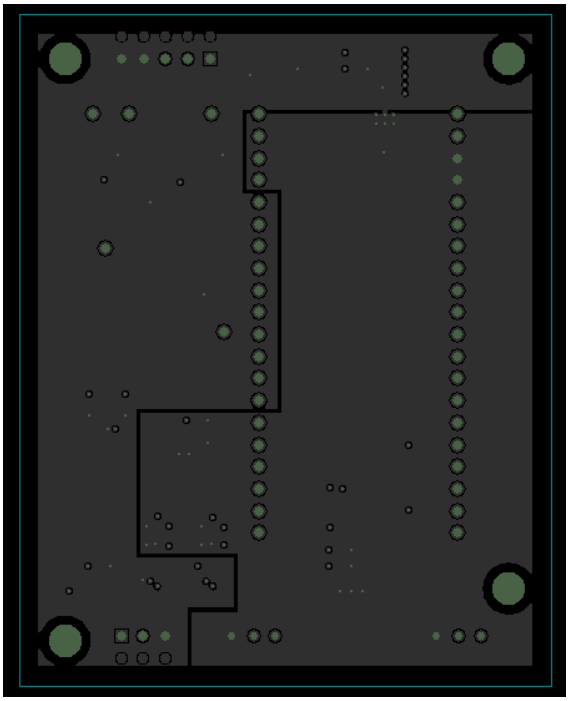
PCB Gerbers



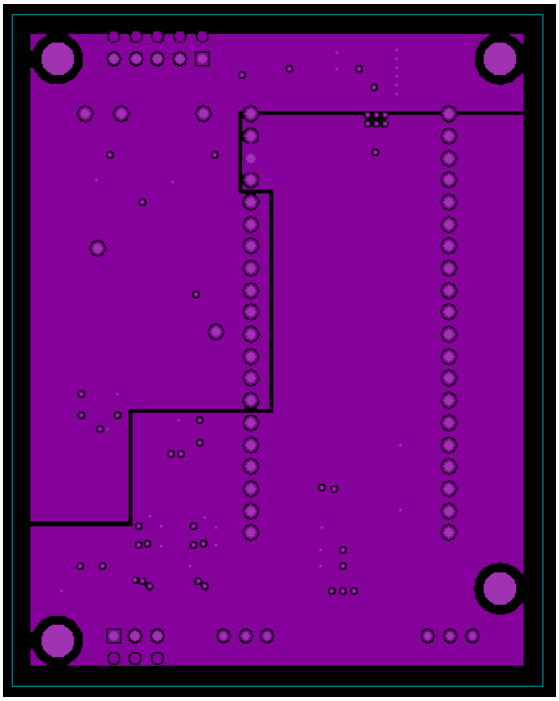
Top plane + Silkscreen



Bottom plane + Silkscreen



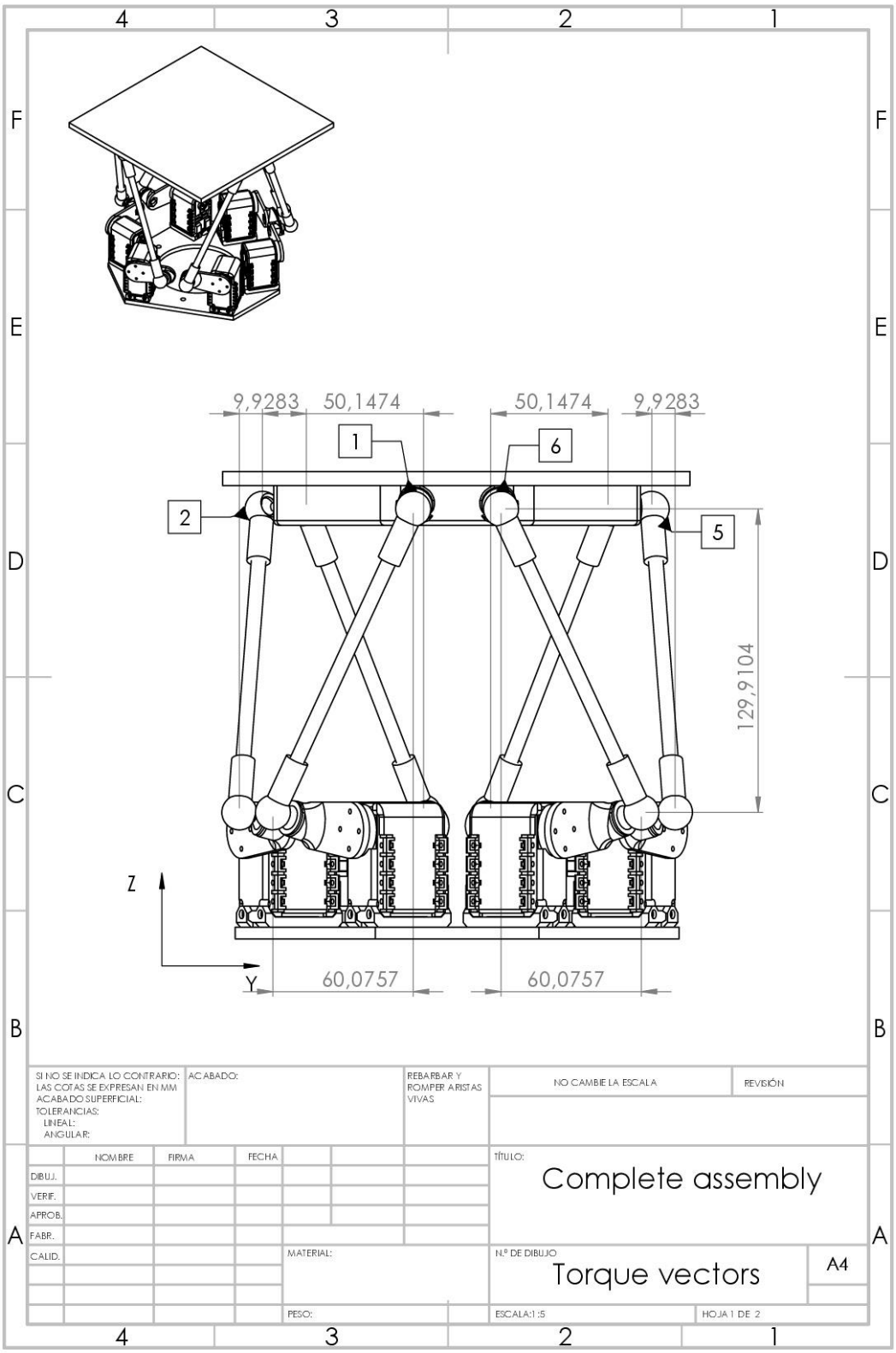
Ground Plane

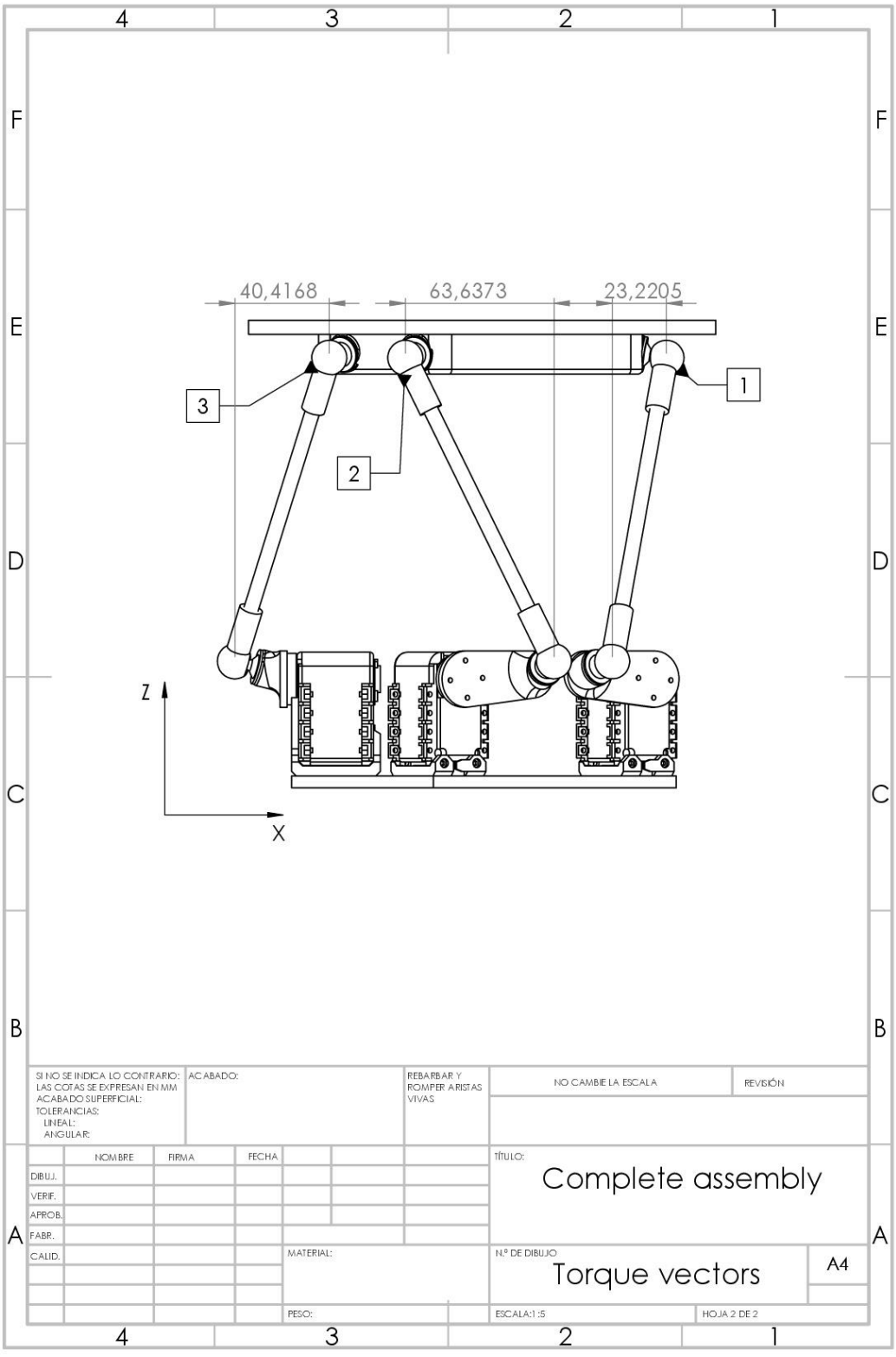


Power Plane

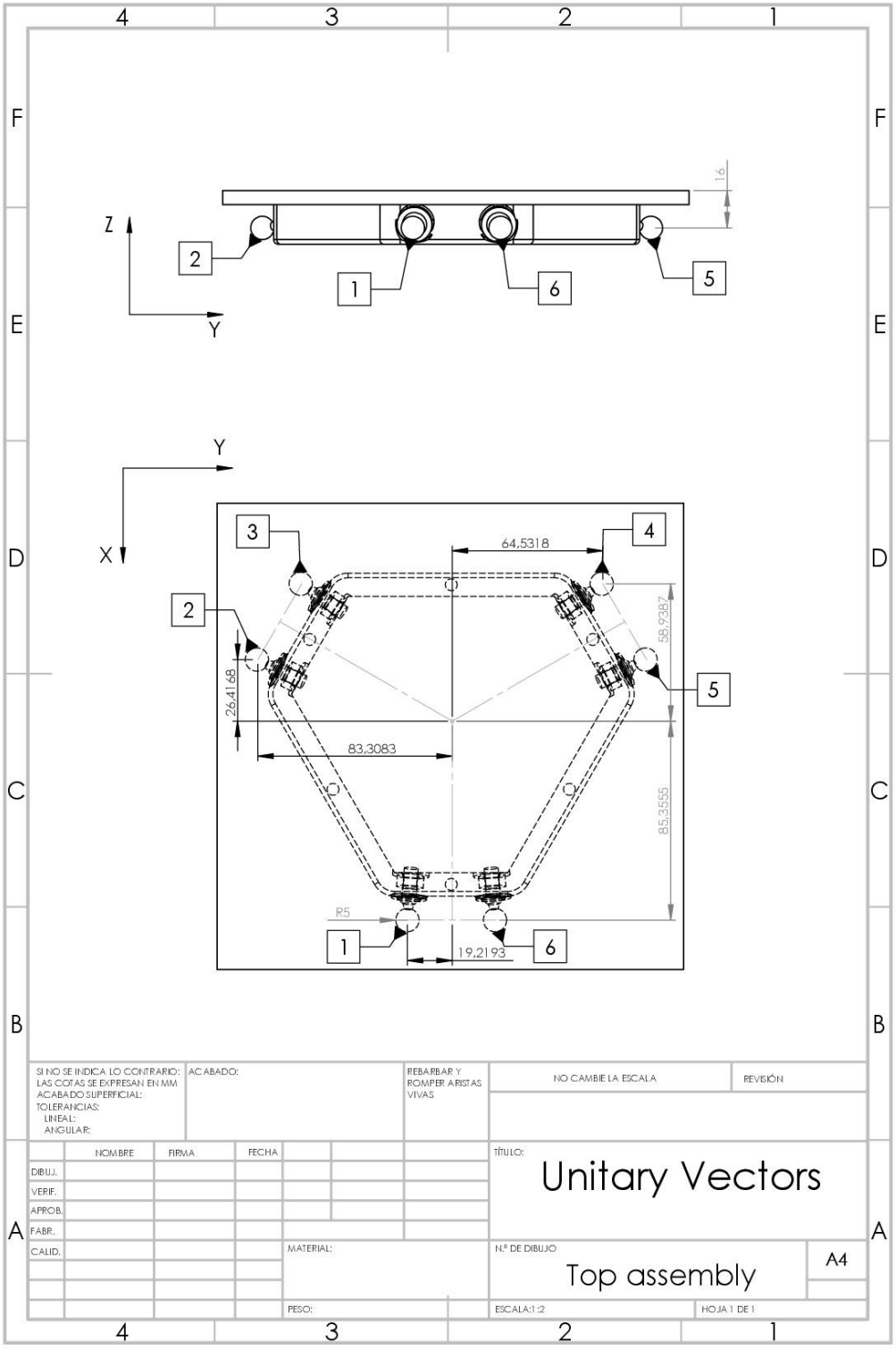
Appendix E

Gough Platform Force/Torque Sensor Measures





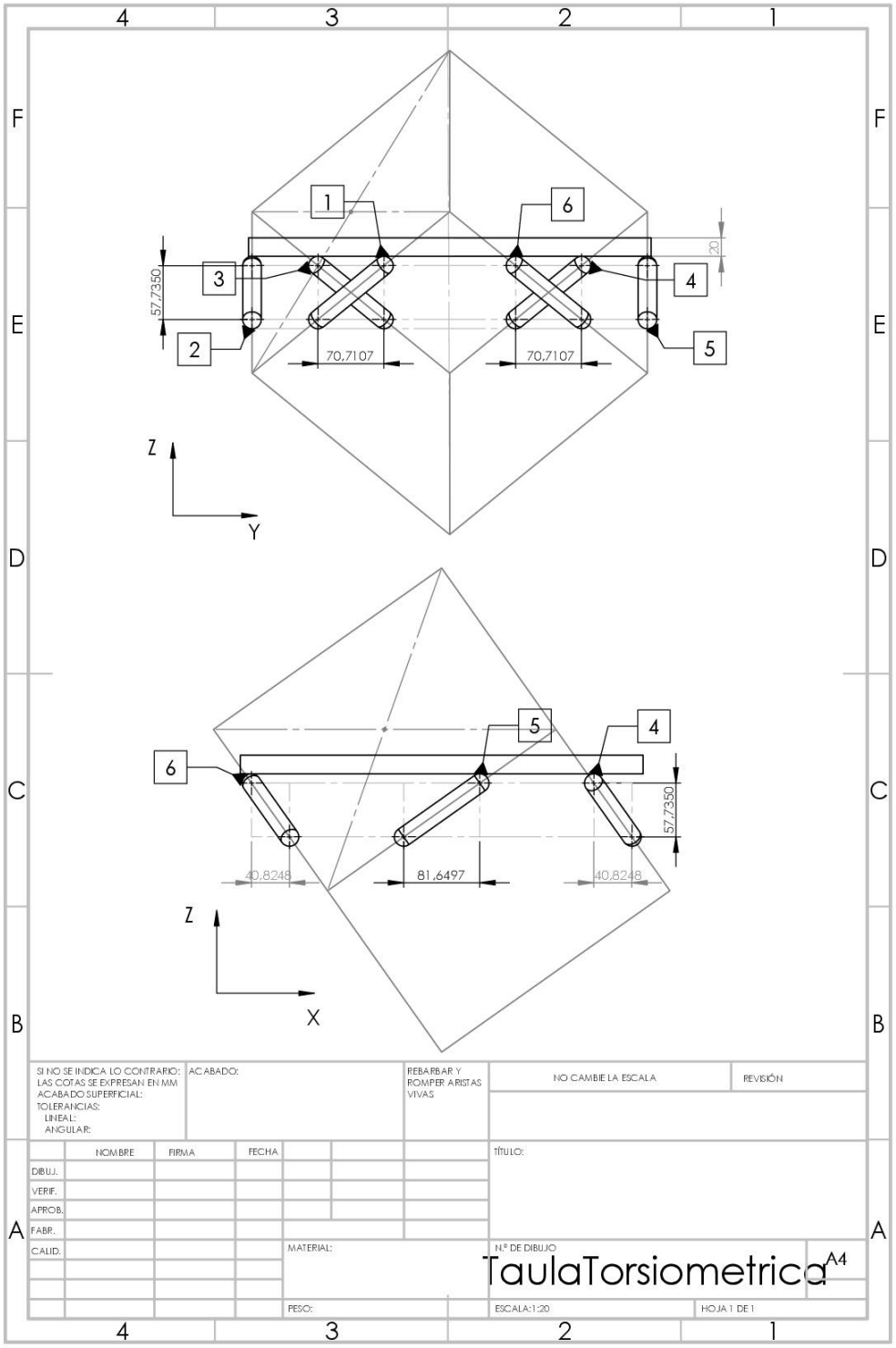
e1 = [-23.2205, -60.0757, -129.9104]
e2 = [63.6373, -9.9283, -129.9104]
e3 = [-40.4168, 50.1474, -129.9104]
e4 = [-40.4168, -50.1474, -129.9104]
e5 = [63.6373, 9.9283, -129.9104]
e6 = [-23.2205, 60.0757, -129.9104]



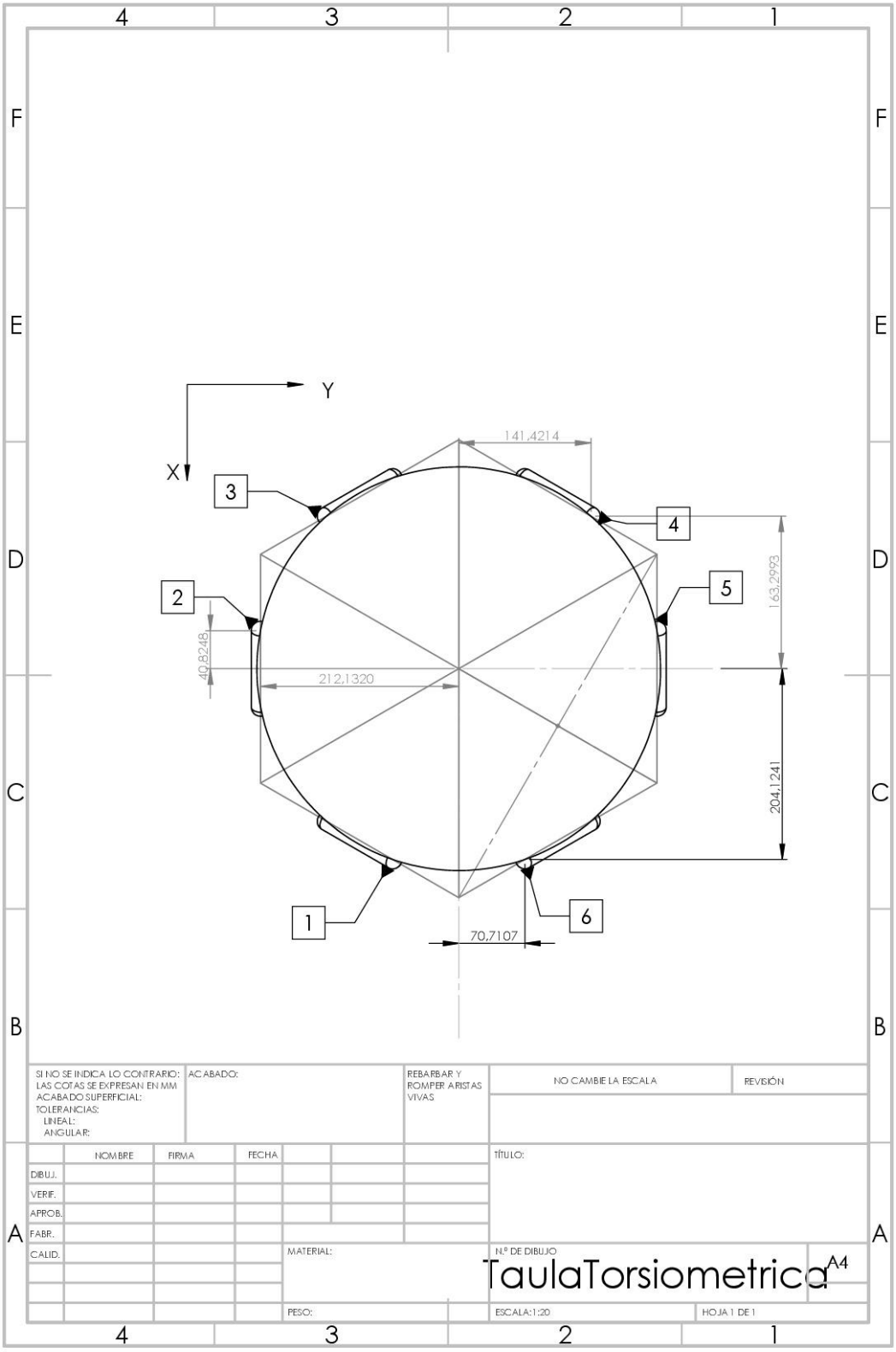
r1 = [85.3555, -18.7765, -16]
r2 = [-26.4168, -83.3083, -16]
r3 = [-58.9387, -64.5318, -16]
r4 = [-58.9387, 64.5318, -16]
r5 = [-26.4168, 83.3083, -16]
r6 = [85.3555, 18.7765, -16]

Appendix F

Wrenchpad Force/Torque Sensor Measures



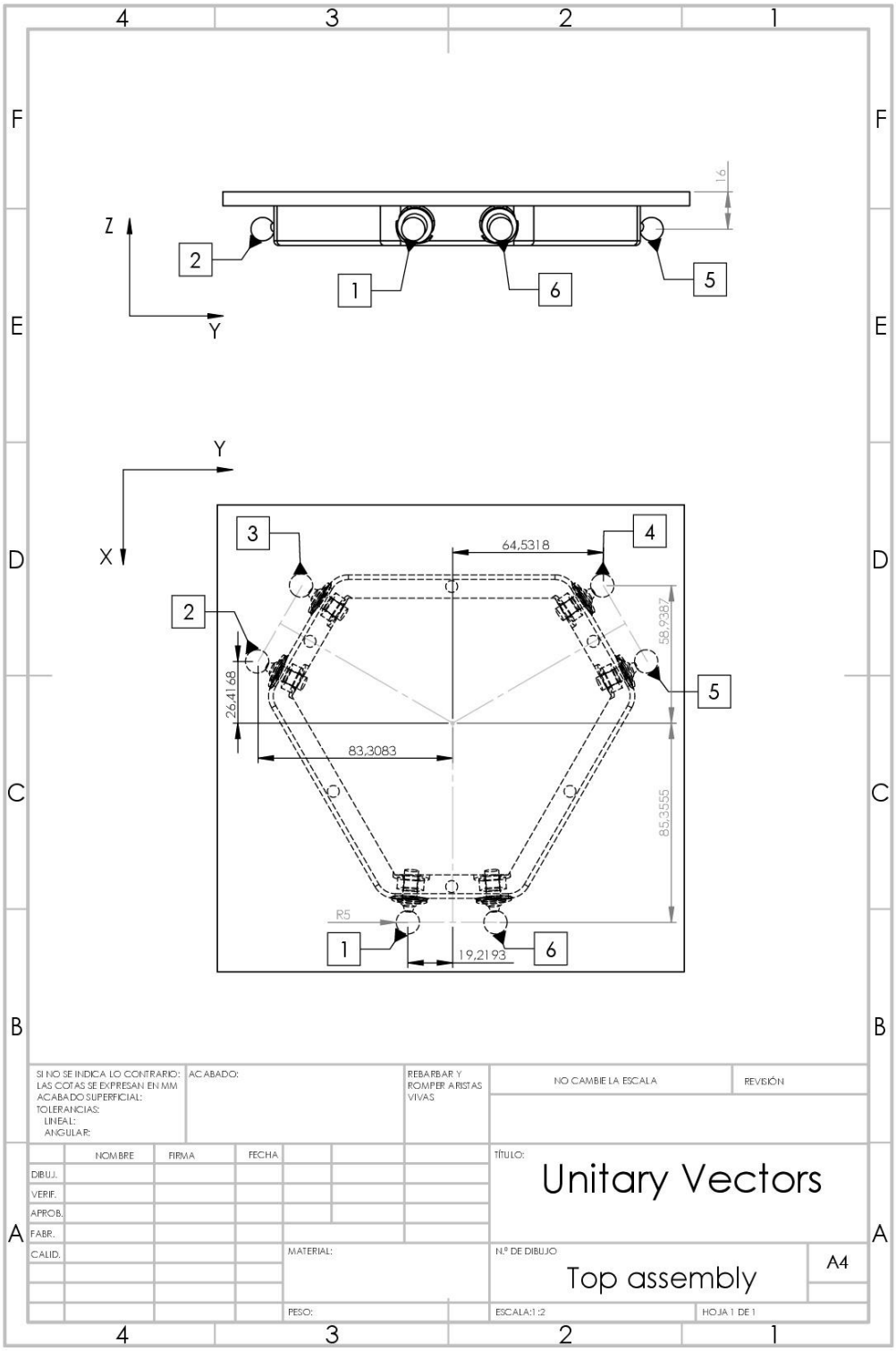
e1 = [-40.8248, -70.7107, -57.735]
e2 = [81.6497, 0, -57.735]
e3 = [-40.8248, 70.7107, -57.735]
e4 = [-40.8248, -70.7107, -57.735]
e5 = [81.6497, 0, -57.735]
e6 = [-40.8248, 70.7107, -57.735]



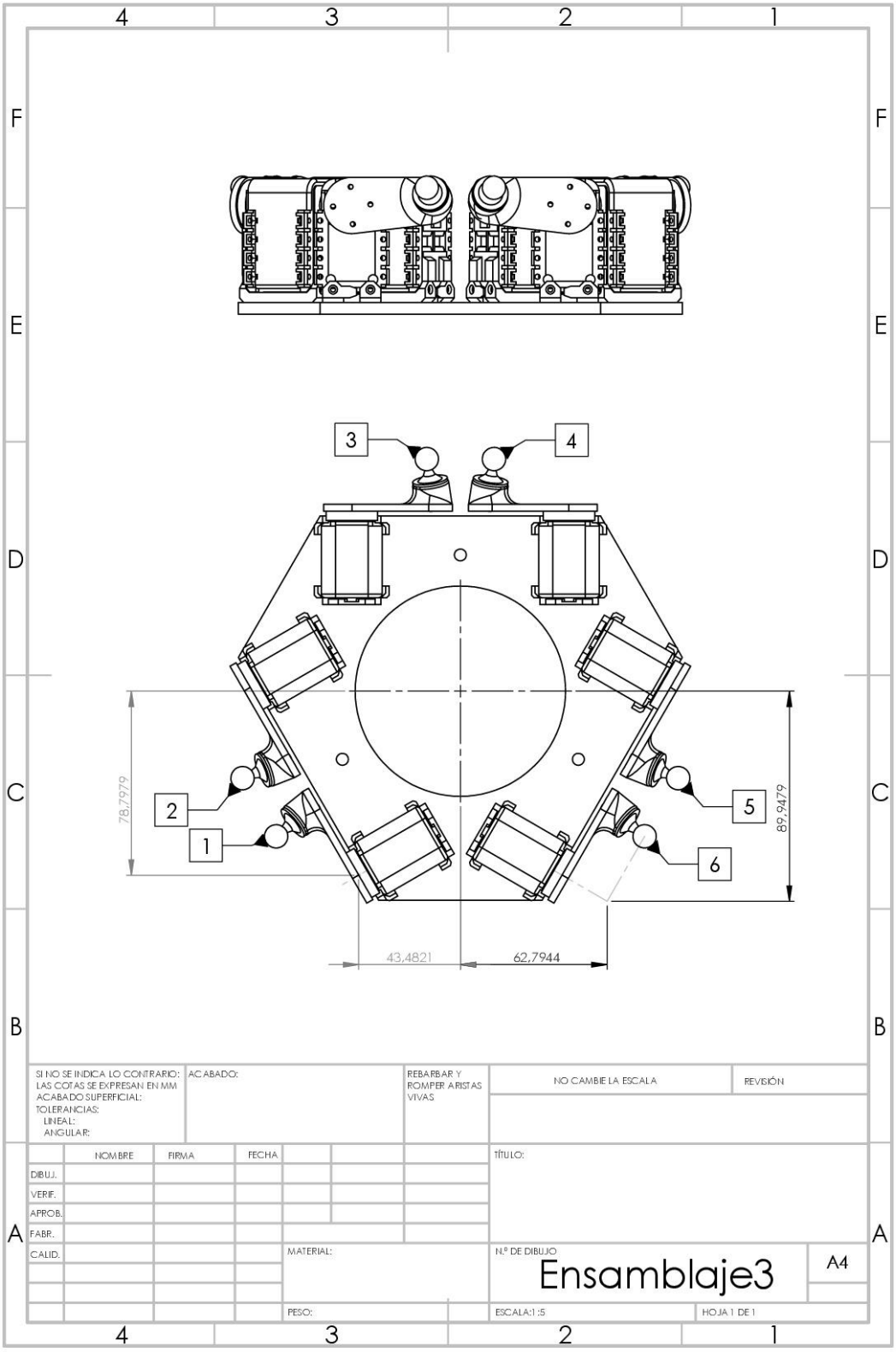
r1 = [204.1241, -70.7107, -20]
r2 = [-40.8248, -212.132, -20]
r3 = [-163.2993, -141.4214, -20]
r4 = [-163.2993, 141.4214, -20]
r5 = [-40.8248, 212.132, -20]
r6 = [204.1241, 70.7107, -20]

Appendix G

Gough Platform Actuation Measures



L1 = 85.3555mm
L2 = 19.2193mm



L3 = 89.9479mm
L4 = 62.7944mm