POLITECNICO DI TORINO

Master's degree in Mechatronic Engineering

Master's degree Thesis

Study of a weight balancing system for an agricultural robot

SUPERVISORS

Prof. Marco VACCA
Simone ALLASIA

STUDENT

Andrea Branca
S276507

A.Y. 2022-2023

Contents

LIST OF FIGURES

Chapter 1

1.1 Introduction: The robotic side of agriculture

As in all other sectors, also agriculture is developing new technologies to help farmers get maximal results with minimum effort. In this way the 4.0 farming was born: the new trend of agriculture to improve the quality of work. The main innovations on the large scale regard, for example, self driving tractors or new kind of spreaders and fertilizers; new technologies are also in the forestry sector. One of these is the new robotic tool carrier, which in the last year has become more popular.

Figure 1: a robotic tool carrier

This type of machines are often equipped with a front mulcher, like in the picture above, but thanks to their versatility they can attach a large variety of tools, starting from a sprayer and arriving also to large forestry stump remover.
The main characteristic of these machines is that they can go in very rough terrain where a normal tractor can not arrive; in particular they do an important work on hill side as they can reach quite important slopes.
One of the most famous brand is “Energreen”, an Italian company which develops the ROBO tool carrier, one of the first robotic tool for agriculture in Italy.
The main characteristic of this robot, is that it uses a remote radio command for driving the machine, this leads to protect the operator from accidents because he can command the machine from 150m of distance in a secure area.
Another strength of the tool carrier, is the capacity of working in a very difficult scenario with slopes up to 55° in all directions, and thanks to their powerful diesel motors they can be equipped with large tools. On the other hand, this motor provides also a large amount of weight that can help stabilize the machine, but may be a problem when the operator needs to move to a different operating site.[1]

Another possible problem of this kind of machines is the price. The average cost is around 60,000€ and for some companies it can be a problem to pay a price like that.
The characteristics of these robots can really help the work on difficult scenarios. The user can stay in a safe zone, far from the danger while the robot is working in a dangerous place.
The combination of this and the power of the motor, is at the base of the success of these machines.
These vehicles can be used in a multitude of different environments. If the “Energreen” is mainly devoted to the cure of the green areas, there is an Estonian company that develops robotics solutions for challenging the climatic changes. The name of this company is”Mirlem Robotics”. One of the models developed by the Mirlem is devoted to planting trees. It has a system of autonomous guide that can plant 1 ha of trees in around 6 hours [2]. Mirlem Robotics provides also some models that are dedicated to the defense, so that they can be applied in war zones, or models that can help the population in case of earthquakes or fires.
The compact dimensions and the light weight are the main characteristics of any of these models.

1.2 Our Project

The goal of this thesis is to create, in collaboration with ALPIrobot of Bagnolo Piemonte (CN), a new type of agricultural robot, which is capable of being attached to different tools and capable of self balance the weight of the machine in a way that both tracks undergo the same weight while working with side slopes.

Figure 2: ALPIrobot's logo

As said before, ALPIrobot is creating this new machine, and my goal was to create the system capable of balancing the weight.
In order to have the best possible grip under all conditions, the machine is mounted on two rubber tracks, this allows to work in a wider range of conditions.
One key point of the ALPIrobot's machine is that each part is easily accessible by the user, helping a lot the maintenance.
Finally the main characteristic of the machine is that when it works on a slope, the motor will shift and rotate in order to move its center of gravity and then distribute more equally the weight so that it won't create a sign of where the track pass.
The final result will be a machine that can be used for several applications, especially in wine yard or fruit yard where spaces are small and the terrain isn't typically flat.
In order to make it, we needed to have a small tractor but with enough power to carry tools, so the company decided to mount a 3 cylinder 40hp motor made by Chinese Yanmar.

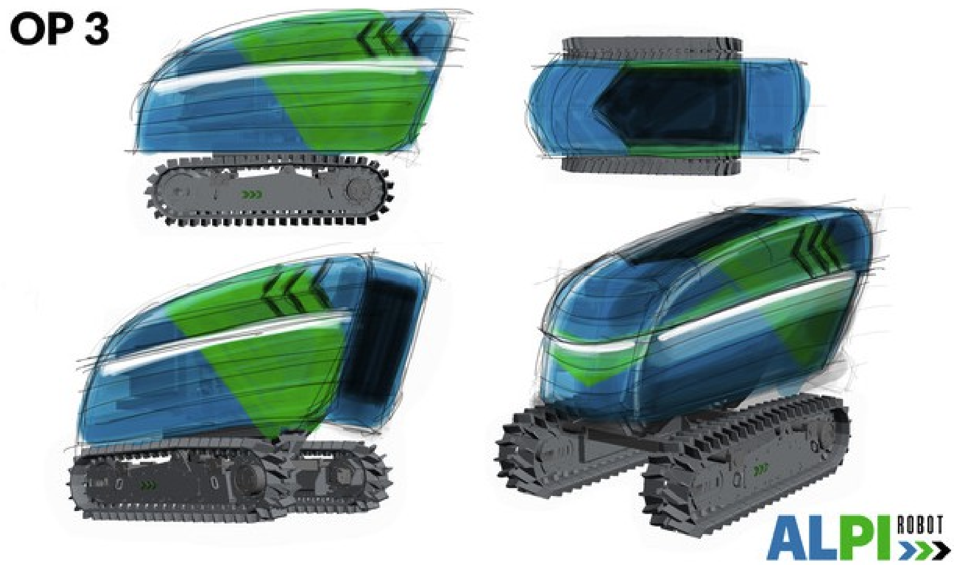


Figure 3: a model of the tractor

In order to pass in the small spaces of vineyards the machine should be very compact, in fact ALPIrobot studied the smallest machine possible, which is 2m long and 1,26m wide.
In the picture above is represented a model of the tractor that I will consider during this thesis.

1.3 State of art

These type of machines, as said before, is becoming more popular in these years and lots of brand are prototyping them.
One of the most famous company is Energreen, that builds lots of models of this type of robot. All of them are radio commanded and they go from nearly 20 hp to almost 100 hp. These powerful machines can be attached to a large variety of tools, from the brushes for cleaning the streets from leaves to the forestry header which can mulch trees or branches.[1]
Another company that produces this type of machine is FAE, a company that produces two models: one of 50hp and one of 70hp that are mainly dedicated to the cleaning of large areas with mulcher.
Lots of companies that produce tractors or machines for the cure of green areas are developing this new type of robot and basically all have the same characteristics:

- Compact dimensions;
- Hydro static transmission;
- Possibility of reaching difficult areas;
- Radio controlled;
- Versatility: they can be coupled with lots of tools.

Each of these machines, anyway, has the possibility to reach very high inclinations: most of them can arrive to 55° of lateral slope and about 35° to 40° in longitudinal slope. None of them has the possibility to attach two tools at the same time, for example a front mulcher for

cleaning a row of a wine yard, and a sprayer attached on the rear to do some treatments on the plants.
The goal of our machine will be exactly this, combining the possibility of reaching the hardest areas but having the capacity to attach multiple tools simultaneously. Then the main goal that none of the machine I saw has, is the possibility to balance the weight in order to stabilize the machine and distribute equally the weight onto the tracks.

1.4 Machine description

The machine that we are going to study in this thesis is the robotic tool carrier developed by the ALPIrobot. This machine is designed to work mainly in fruit-yards or wine-yards but can work also in larger areas where normal tractors can't go (for example in the woods under the trees or on very steep areas like the sides of hills or mountains).
The machine is equipped with a 40hp diesel motor, made by Yanmar. At full weight, so with two tools attached to it arrives to weight around 1600kg. On the front and on the rear there are two hitches where it can attach some tools. In particular, it is studied to carry easily a mulcher on the front and a sprayer on the rear. The weight each hitch can lift is around 300kg.
The machine is radio controlled so the operator can maneuver it from a safe place. All the actions made by the machines are controlled by a PLC that works as head of the machine.
Our robot is equipped with two rubber tracks in order to have more grip on the soil in all conditions and distribute more equally the weight on the terrain.
As said before, the goal is to have an equal distribution of the weight in all conditions: the machine has the possibility to move its heaviest part (the motor) in order to move its gravity center and then to spread the weight on both tracks.
The machine can be represented with this simple scheme:

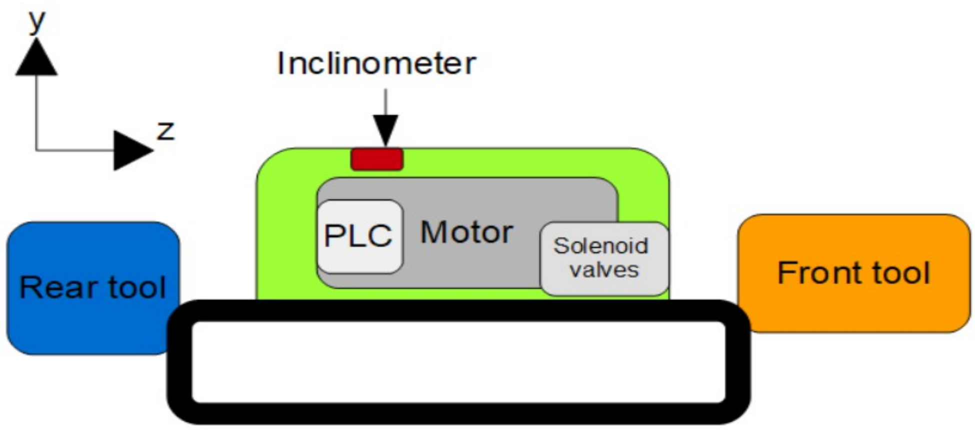


Figure 4: simple block scheme of the machine

As it can be seen, in red it is highlighted one of the sensors used to control the position of the motor: the inclinometer measures the inclination of the machine every 10 ms.
The PLC (the brain of the machine) is positioned on the side of the machine so it can be easily

reached by the user in case of necessity.
The solenoid valves that drives the movement of the motor are positioned in the front of the machine, next to the oil pump and the oil tank.
The motor obviously occupies most of the space and produces a lot of heat; it is important then to have a good cooling system and also have devices that can work with quite high temperatures.
In a simple model, the machine can be divide in two mechanical parts:

- The motor;
- The chassis.

If necessary there are also the tools but for now let's just consider these two.
The motor weights 650 kg and the chassis weights 500 kg .
Since the chassis has very low center of gravity, at around 200 mm from the soil, the difference of weight that it is created in the worst condition is very low compared to the one generated by the motor; so the chassis will be considered as it always distributes equally the weight on the tracks.
The motor weights 650 kg and its center of gravity is at 820mm from the ground, this will create a large difference on the weight distribution that will be compensated by using some pistons to rotate and translate the motor on the chassis.
The peculiarity of this machine is the capacity of moving its center of gravity in order to equally distribute the weight on both tracks in all conditions. As the machine senses an inclination the motor block moves, in particular it can rotate and translate.
The system that controls everything is governed by the PLC and it is basically formed by two position sensors, the inclinometer and two pistons that are the responsible of the two movements. A more detailed description is present in the chapter 5.

Chapter 2

2.1 The starting point

In order to understand correctly how to balance the weight, we have to study its distribution on the tracks.
To do so, I assumed that a track can be represented by a fixed hinge since it can be assumed that it blocks all movements over horizontal and vertical axis and the momentum along the third axis.
Here is represented how I modeled the track:

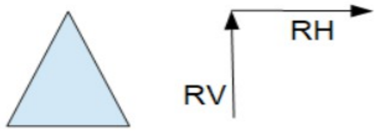


Figure 5: constraint reaction of the track

With this assumption, when the machine operates on flat areas, the weight is equally distributed over the two track as shown in the following figures:

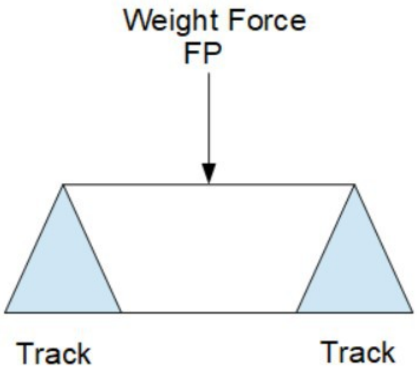


Figure 6: machine simple representation

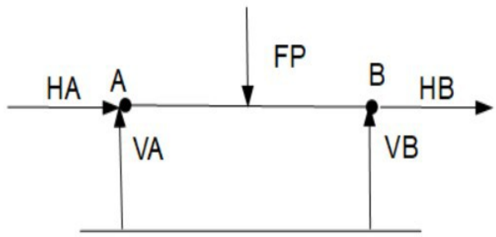


Figure 7: all the forces acting on the machine in a flat plane

The whole system is static so, thanks to the Fundamental Equations of Static, the sum of all forces on the vertical axis, the sum of all forces on the horizontal axis and the sum of all momentum has to be zero. Then in the flat plane condition we will have:

$$F_P = V_A + V_B \quad (1)$$
$$H_A = H_B \quad (2)$$
$$V_B * l = F_P * \left(\frac{l}{2}\right) \quad (3)$$

By doing some calculations,we can easily say that Fp will be split in Va and Vb which are both equal to $\frac{Fp}{2}$.

As we start to change the operating conditions and we start to operate on slopes the Weight force Fp is split into two components: one parallel to the plane and one orthogonal. With the presence of this new orthogonal component, now we don't have the equilibrium anymore, like on the flat plane. Now the vertical reaction will change with the change of the slope.

$$F_v = F_p * \cos(\alpha) \qquad (4)$$
$$F_h = F_p * \sin(\alpha) \qquad (5)$$

Since I examined a static condition I assumed that the horizontal component of the reaction will compensate the static friction force that is present between tracks and soil, so

$$H_A = F_s \qquad (6)$$
$$H_B = F_s \qquad (7)$$

where F_S is the friction force.

Then the formulas 1, 2, 3 now, in inclined plane become

$$V_A + V_B = F_V \qquad (8)$$
$$F_H = H_A + H_B \qquad (9)$$
$$V_B * 2 * d = F_V * d - F_H * h \qquad (10)$$

from (10) we can obtain the new value for the vertical reaction in the point B that is

$$V_B = \frac{F_V * d - F_H * h}{2\text{d}} \qquad (11)$$

Now that I have this result, in order to have the same weight on both tracks we have to set

$$V_A = V_B$$
$$V_B = \frac{F_V}{2}$$

In order to have this compensation, the machine can move in two ways:

1. Rotate the whole motor up to 25°;
2. Translate of 105mm the motor.

With these two movements the machine's center of gravity will change its position and then also the point where forces are applied. In particular it will shift to left or right of a quantity Δd and can reduce its height of Δh.

Now let's study what will happen if we first translate and then rotate the motor.

2.1.1 Translation phase

In this part I considered only translation.

I know that the maximum translation possible is 105 mm left or right starting from the center,in this particular case the center of gravity will only shift, then the (10) become

$$V_b = \frac{F_V * (d + \Delta d_{trasl}) - F_H * h}{2 * d} \qquad (12)$$

Vb as from hypotheses should be equal to $\frac{F_V}{2}$ Then by substituting it in the (12) and doing calculations I obtain:

$$\frac{F_V}{2} = \frac{F_V * (d + \Delta d) - F_H * h}{2 * d} \quad then$$

$$\frac{F_v}{2} * 2 * d = F_V * (d + \Delta d) - F_H * h$$

$$F_V * \Delta d = F_H * h \qquad (13)$$

from last equation and using the (13) we can calculate the angle that we can compensate with only translation; in fact: $\frac{F_V}{F_H} = \tan(\alpha) = \frac{\Delta d}{h}$ and then if I use Δd_{max}, the maximum angle will be discovered $\frac{\Delta d_{max}}{h} = \tan(\alpha_{max}) = \frac{105}{520}$ *then* $\alpha_{max} = \arctan(\frac{105}{520}) = 11.41°$.

So, if we only translate we can compensate a maximum of 11,41°. To compensate more we need to add the rotation.

2.1.2 Rotation phase

For this phase we will follow a similar reasoning of the translation phase, we study how the height and the lateral translation of the center of gravity will change.

Here both height and lateral position change, in particular they become:

$$h_{new} = h * \cos(\theta) \qquad (14)$$

$$d_{new} = d_{start} + \Delta d_{tra} + \Delta d_{rot} \qquad (15) \quad with \quad \Delta d_{rot} = h * \cos(\theta)$$

I considered that this rotation is added to a previous translation, then following the formula of before we have

$$\frac{F_V}{2} = \frac{F_V * (d + d_{trasl} + d_{rot}) - F_H * h_{new}}{2 * d} \qquad (16)$$

$$F_V * (\Delta d_{trasl} + \Delta d_{rot}) = F_H * h_{new} \qquad (17)$$

from the last equation, we can calculate again an angle by doing some arrangements. In particular we know that

$$F_V = mgcos(\alpha) \quad and \quad F_H = mgsin(\alpha)$$

then using the formulas d new and h new we obtain:

$$mgcos(\alpha) * \Delta d_{trasl} = mghsin(\alpha)\cos(\theta) - mghsin(\theta)\cos(\alpha) \qquad (18)$$

by simplifying mg and using trigonometric formulas we obtain

$$\Delta d_{trasl} * \cos(\alpha) = h * \sin(\alpha - \theta) \qquad (19)$$

$$\sin(\alpha - \theta) = \Delta \frac{d}{h} * \cos(\alpha) \quad then \quad \theta = \alpha - \arcsin\left(\frac{\Delta d_{trasl}}{h} * \cos(\alpha)\right) \qquad (20)$$

To get the maximum possible compensation we have to set the d_{max} and the theta at maximum, then using the formula (17) we obtain

$$mgcos(\alpha) * (105 + h * \sin(25)) = mgh * \cos(25) * \sin(\alpha)$$
$$\tan(\alpha) = \frac{105 + h * \sin(25)}{h * \sin(25)}$$
$$\alpha = 34{,}57°$$

so if we add the rotation we reach 34,57° of lateral slope and even in this condition we can have almost the same weight on both tracks.
Now I can use a different approach to study this compensation: we can first rotate and then translate the motor.
The first rotation here will be simpler because we have just to rotate the motor of the same degrees as the slope, while before we had to balance the weight. So here let's just put $\alpha = \theta$.
For the translation we have again the equation (17), so we can do the same passages and in the end obtain again that the maximum angle where the machine can compensate the slope is 34.57°.
With this second method we have the advantage of covering a larger slope (25°) with the first rotation, so we can move less the machine in not steep areas.
The main problem of doing one movement with respect to the other is that the machine, due to its compact dimensions, have very narrow space, so moving the motor completely tilted may risk to touch the tracks. To avoid this kind of problem we decided to operate this compensation in an hybrid way that will be explained later in detail but basically consists in translating and then rotating at the same time.

2.1.3 MATLAB simulation

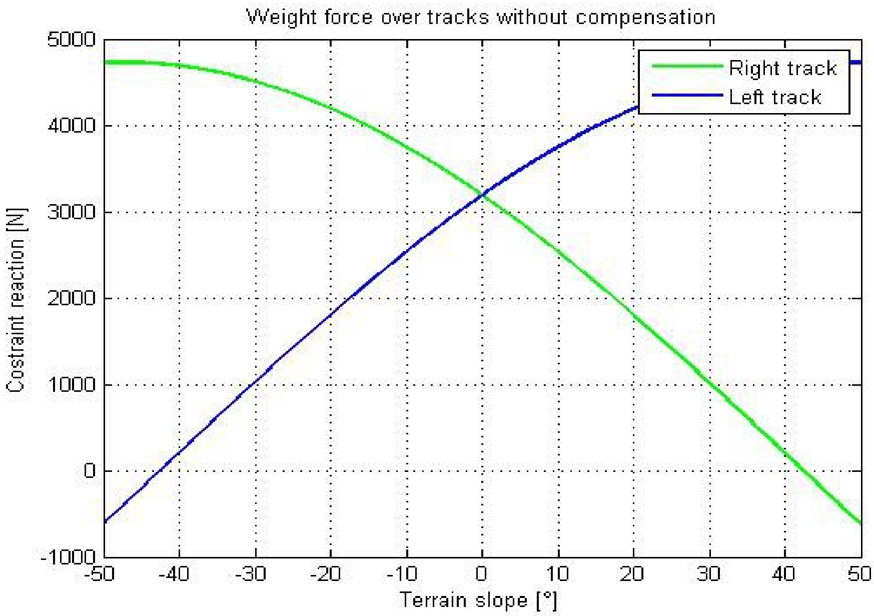


Figure 8: Constraint reaction when no compensating action is done

In order to have an idea of what will happen to the motor when the machine is in motion, I created some graphs of the reaction of the track with different angles.

The picture above represents the reaction of tracks as function of the inclination: as you can see, when the machine is on the flat plane (0°), both tracks experience the same force, then if the terrain starts to be on a slope the reaction discrepancy become larger as the slope increases.

Then what happen in the case of compensation is shown in the three following figures. As you can see in all the pictures there is a flat area. In that zone with the previous calculations we have exactly the same weight and then the same reaction on both tracks. As expected the rotation covers a bigger area with respect to the translation.

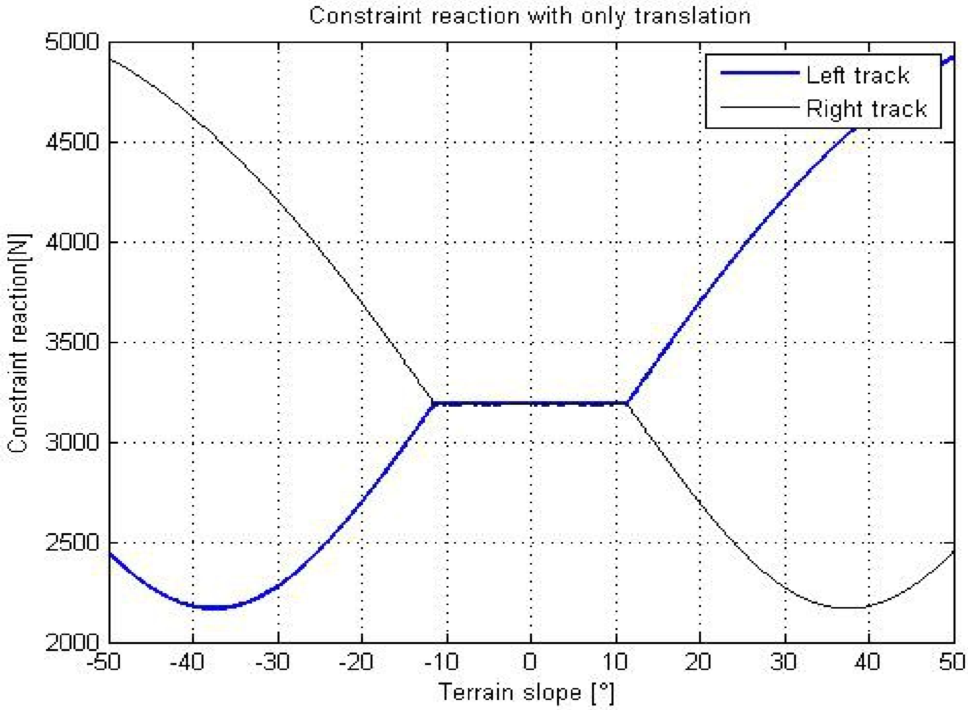


Figure 9: constraint reaction when we operate with only translation

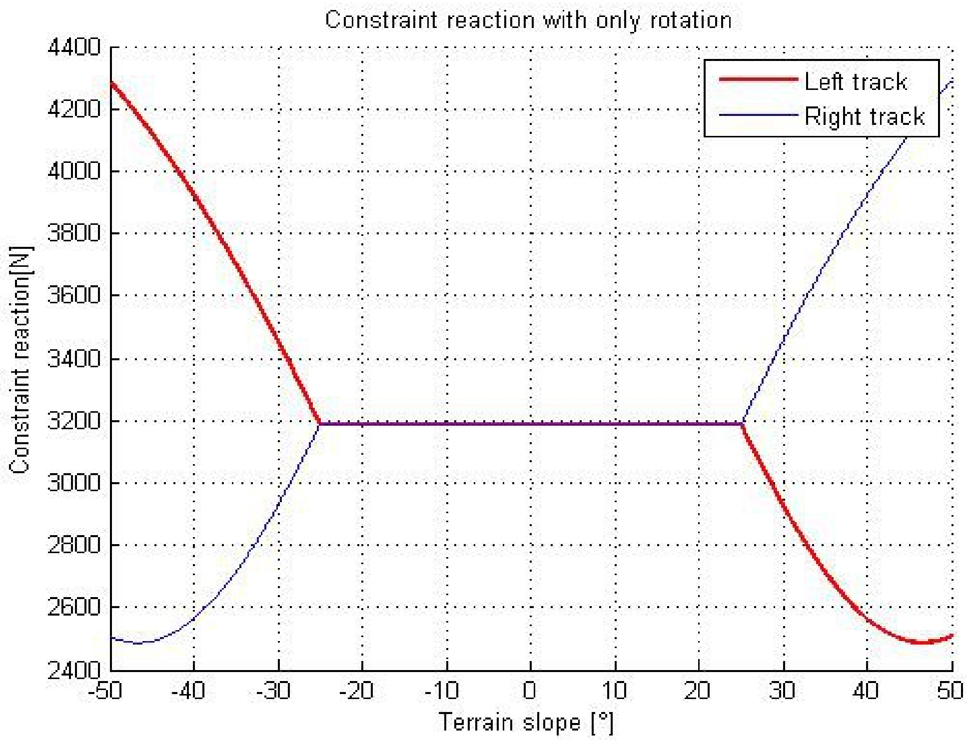


Figure 10: constraint reaction with only rotation

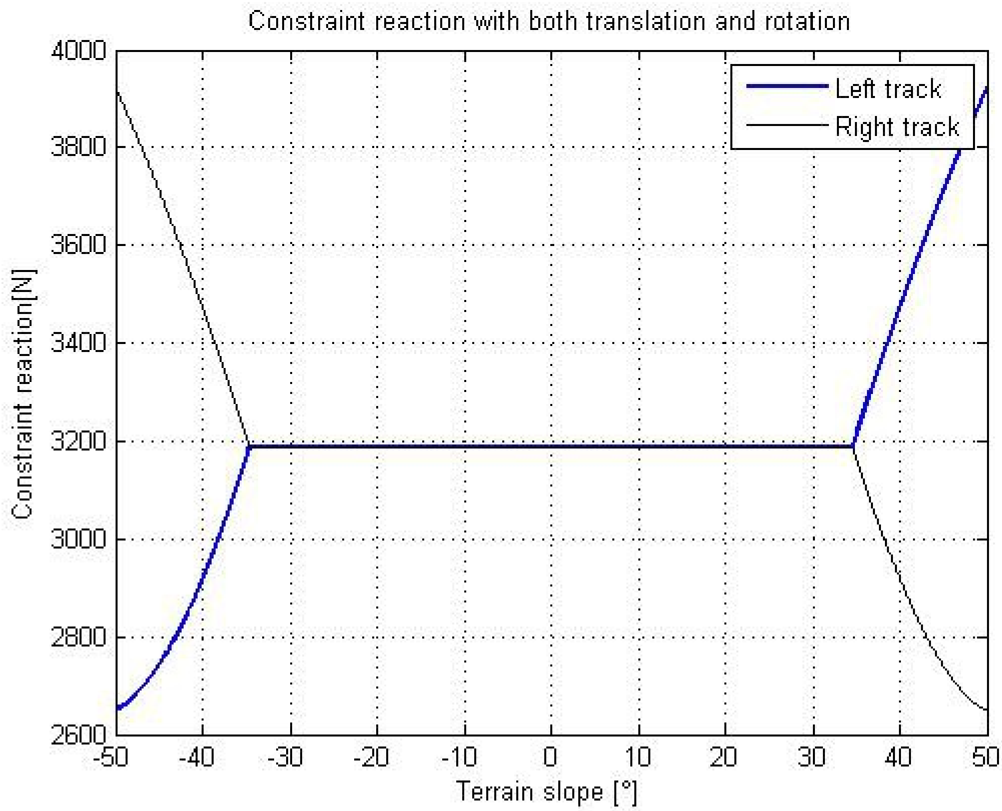


Figure 11: constraint reaction with all the possible compensations

Chapter 3

3.1 Electronics

3.1.1 PLC

The core of the machine is the PLC. PLC is an acronym that means Programmable Logic Controller and it is basically a kind of control unit (CU) that can be used in a large variety of environments, they were born to serve industrial processes. Nowadays they found application in almost every sector, from the automotive to the industry, controlling production lines or controlling robotic arms.
Since the '70s, they were fundamental for industrial development,when they were used to help manufacturers to control production lines. However they were made by relays and so the control was tricky and complex. With the progress of technology and the development of powerful algorithms, PLC became easier to program and everyone could do it.

3.1.2 How to program a PLC

As said before, the first PLCs were a bunch of relays put together and by opening or closing the contact inside of them they did a specific action on the line where they were inserted. With this in mind, we can imagine a process divided into three parts:

- THE INPUT: the signals that enter in the system then in the relays
- THE CONTROL: the network of relays
- THE OUTPUT: the signal that exits from the relays

Relays can be put together to create some logic functions but we have at the most one signal for relays at input and one at output. This “network” is organized in lines and each line creates a rung and the energy flows from left to right and from top to bottom. This way of programming is called LADDER DIAGRAM.
Another way to program a PLC is with the Structured Text (ST) that is similar to programming in C or any other programming language. In fact we have to describe with text what we want to do in the program, so by putting together functions and variables we create a program.
In order to do so, there are IDEs (Integrated Development Environment) that can help us to write what we need, in particular in this thesis I used CODESYS 3.5 to develop the program needed for our application.

3.1.3 How a PLC works

The main goal of a PLC is to take data from sensors, elaborate them and provide signals to actuators.
A PLC in simple way is composed by three components:

- INPUT MODULE: receive all the signals needed by the sensors connected to it
- OUTPUT MODULE: provide the signals to the actuators
- CPU & MEMORY: CPU elaborate the input signals to obtain the outputs, while in the memory are saved the code and variables.

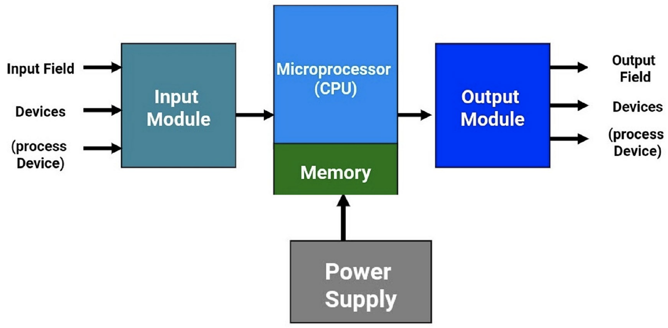


Figure 12: block scheme of a PLC

(Towards generic memory forensic framework for programmable logic controllers,(Rima et al, 2023)[3])

The PLC works in a cyclic way: it executes the code by taking the data from input module, then puts them in the code and provides the output; then the cycle restarts.(Awad,2023)[3]

3.1.4 IFM CR720S

In our project we decided to use the ifm 720 PLC since it was the one that has most

Figure 13: the IFM CR720S controller

https://www.ifm.com/it/it/product/CR720S?tab=documents[4]

characteristics that we want. In particular it is intended for mobile machines and can be easily programmed with CODESYS.
As we can see it has several connectors on the front: four are circular and two are rectangular.
The four circular connectors (M12 Connectors) are devoted to the data transmission, in fact we have four can interfaces, one RS232 serial communication interface and one the Ethernet interface.
In total we have 98 between inputs and outputs both analog and digital, this allows us to work with lot of signals and connect several sensors to it.
Another important characteristic of the 720S is the presence of two PLCs inside of it, one dedicated to the so called SAFETY application and one to the standard application. This separation allows the programmer to separate the normal functions of the controller from the functions that are relevant for the life of the machine and from the functions that help the secure failure of the code.[4]

3.2 Communication

Another crucial aspect of the whole system is the communication. As said before, the PLC provides different possibilities to communicate, the main protocol that is used in this project is the CAN-open protocol.

3.2.1 CAN protocol

The control unit of the machine is the PLC.
In order to have control over all the machine the PLC has to know exactly what it is happening in every part of the machine. To do so it has to receive lots of data from sensors and it has to transmit other signals to the actuators. Some sensors can be directly controlled but there are other sensors that need to use the CAN interface or the serial port to communicate with the PLC.

Figure 14: CAN open logo

The CAN-open protocol is based on the CAN-bus, a protocol that establishes rules for the communication between objects. CAN stands for Controller Area Network and it represents the set of devices that are connected together to form a network.
The CAN-bus protocol covers only the first two layers of the ISO-OSI stack: the Physical and the Data Link Layer. At physical layer we have the description of the connection link in terms of type of cable, nodes requirements and electrical characteristics of the signals and of the cables. At data link layer, the communications requirements in terms of baud rate, cable length and cable terminations are explained .
The base of the CAN-bus protocol are the two wires that pass through the whole system where there are the nodes that may communicate between each other. These two wires are called CAN_H and CAN_L.
This particular type of physical network was developed in order to achieve robust

performances in every conditions, even in the worst.
The nodes are the devices, equipped with a CAN interfaces that belongs to the bus, the bus is formed by the two wires mentioned before. Since the signal travels through a physical cable, we have maximum transmission speed when the nodes are near and the cable is short. The maximum speed available is 5 Mbit/s with the protocol CAN FD. To transmit correctly at the end of each bus, a resistance of 120Ω is required.

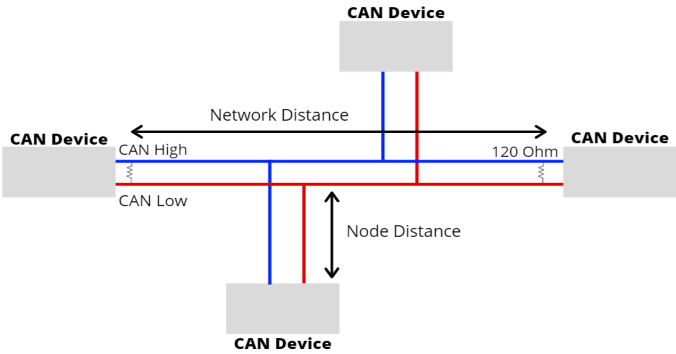


Figure 15: a CAN-bus example

(from: https://www.seeedstudio.com/blog/2019/11/27/introduction-to-can-bus-and-how-to-use-it-with-arduino/)[5]

The main advantages of this type of communication are:

- Robustness: this system is ideal for safety applications due to reliability of the system and thanks to several mechanism to detect errors;
- Low cost: the idea behind the CAN bus is to have the minimum number of errors, less weight and few wires;
- Speed: the signal passes only through two wires then it is received by a device;
- Flexibility: the network can easily change its form by adding and removing nodes since the protocol is well known;
- Eficiency: when the channel is contended by multiple messages the one with higher priority will be transmitted.

So the communication between two users over the bus can be very simple thanks to these characteristics to which we can add the low number of cables and the ability to recover error messages.
The CAN bus protocol can be used in different types of networks creating a secondary division of protocol. In particular:

- High speed CAN bus where speed can reach 1 Mbit/s and it is the base for CAN-open and j1939 protocols;
- Low speed CAN bus whose speed can be up to 125kbit/s and can allow errors;
- Automotive Ethernet have large bandwidth in order to achieve big quantity of data transferred since it is used to provide driver assistance;
- CAN FD: the newest protocol that is designed for high data transfer.

3.2.2 CAN-BUS Frame

The transmission of the messages on the bus is differential between two logic states:

- Dominant: is the logic state when the differential voltage between CAN_H and CAN_L is higher than a threshold, so it is like setting a logic '0' on the bus;
- Recessive: is the exact opposite of before. The differential voltage is lower than a threshold so that on the bus is set a logic '1'.

Over the bus are transmitted frames that can be of two types: the standard data frame and the extended data frame. These two types are almost identical except for the fact that the extended frame has a 29 bit identifier instead of the 11 of the standard. This allows to have different priority levels while working with the extended frame.
These frames are represented in the figure below:

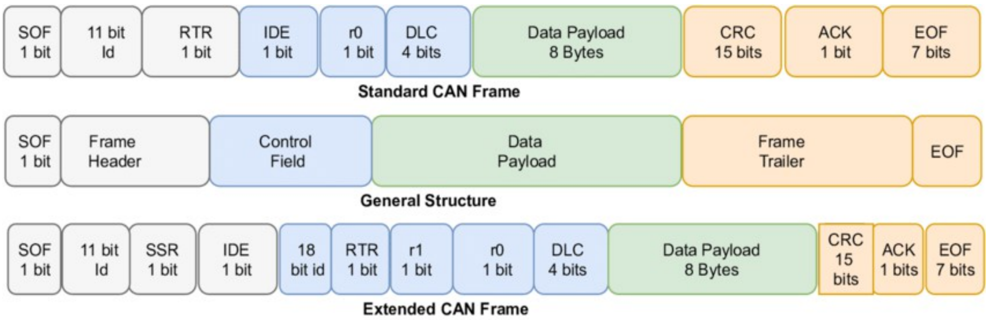


Figure 16: CAN frame description

(from: "Cyberattacks and Countermeasures For In-Vehicle Networks", E.Aliwa,O.F.Rana,C.Perera,P.Burnap, 2020[6])

As can be seen, the message is composed by some elements each one with its purpose:

- SOF: Start Of Frame, tells that a new frame is starting;
- ID: tells the priority of the message;
- RTR: Re Transmission Request is a bit that is set dominant when a transmission is failed and a user wants to try to receive that message again;
- IDE: Identifier that is dominant when a standard frame is transmitted;
- r0: a reversed bit not used;
- DLC: that indicates the length of the data that will be transferred;
- Data Payload: the true data that is transmitted;
- CRC: a checksum to see if the data is received correctly;
- ACK: if the data is received correctly it is set to DOMINANT;
- EOF: is a 7 bit delay interval to avoid overlaps of communications.

In the extended data frame there is also:

- SSR: Substitute Remote Request act like a place holder;

- r1: reverse bit.

3.2.3 CANOPEN

On the CAN-bus basis the CAN-open protocol was developed.
By looking at the stack ISO OSI it can be seen that the CAN bus provides physical an data link layer while the CANopen protocol covers the higher layers.
The CAN open communication is a broadcast communication and messages are exchanged in real time.
This protocol doesn't work as the classic MASTER/SLAVE protocol. All the elements in the network belongs to one of the standardized communication object provided by the protocol.
The CAN protocol also includes:

- SDO:Service Data Object that is composed of two data frames with different id that enable access and then the possibility to change the entries of the object directory, which is the interface between the communication interface and the software application;
- PDO: Process Data Object is composed of one frame the is used to broadcast high priority control and status information;
- NMT Network Management is used to control the state of the devices;
- Error Protocol that checks if every node is "alive" by broadcasting an heartbeat message.

3.2.4 CANOPEN frame

The frame is based on the CAN bus frame, and it is composed by a COB-ID formed by a 4 bits function code, 7 bits of CAN id that limit the number of users to 127, then the RTR and finally the DLC followed by the data.

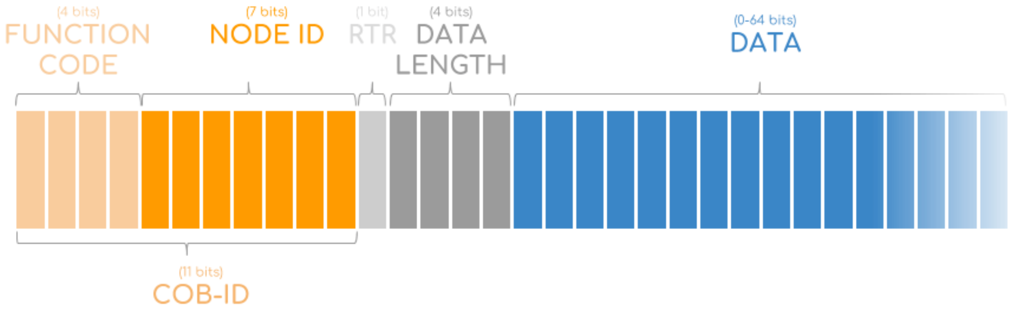


Figure 17: the CAN open frame

(from: M.Falch,"canopen-tutorial-simple-intro",2022 [7])

In order to avoid error on the bus the COB ID is unique.
The 4 bits dedicated to the function code will explain which is the role of the node in the

network. In particular it can be a NMT, a synchronization, an emergency, an heartbeat to control nodes, a time stamp or a PDO/SDO transmitter or receiver.(2023, Velichkov)[8]

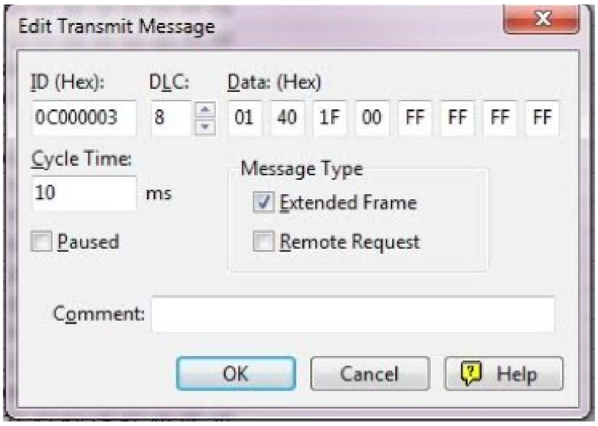


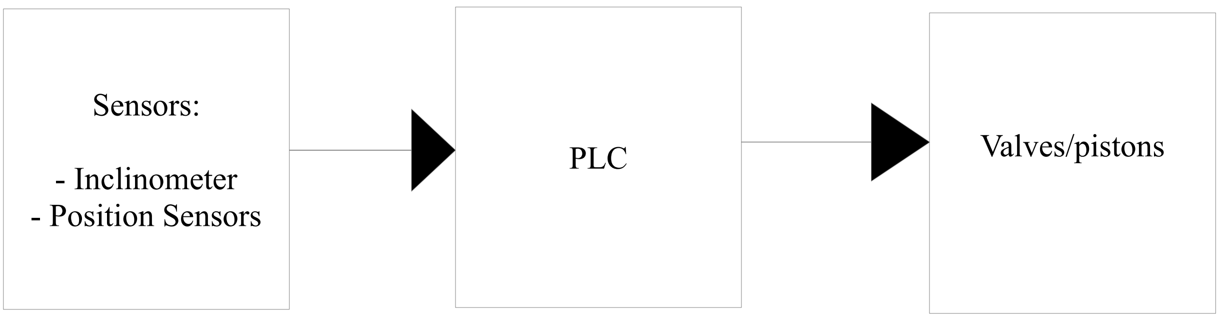
Figure 18: example of a can message

3.3 Block scheme of the machine

The machine is composed of two parts: one mechanical and one electrical. The mechanical part is divided into one fixed part and one mobile part.
The electrical part is composed of:

- PLC that can be seen as the brain of the machine;
- Sensors for inclination;
- Sensors for position of the motor;
- Solenoid valve for driving the piston.

The electrical and the mechanical parts obviously talk to each other.
In the electrical part the sensors measure the quantity needed then they pass it to the PLC that elaborates the data and provides the right signal to the solenoid valve that pilot the pistons. This system can be represented in this way:



Going a bit more in detail, the inclination sensor measures the inclination of the machine and then of the terrain on which the machine is working, then it provides this information to the PLC. If the inclination measured is more than the threshold, the PLC will activate the solenoid valves. Then again the inclinometer measures the inclination and the process starts again.

To explain better the measurement process I reported in the picture 19 a more detailed block scheme.
As it can be seen the PLC is the real center of this machine, it carries out a lot of work. It has to read at every cycle the output of the sensors and provide the new command for the pistons. Then we have the two separated paths for the inclination and rotation of the motor block. The feedback on the position sensors is controlled by the PLC itself.
Even if there isn't a PID that directly controls the position of the motor, the algorithm written in the PLC runs at a speed so that the possible events that can happen don't affect the positioning of the motor. Another reason why the PID is not inserted in this moment it is because, for now, we don't need high precision in the positioning.

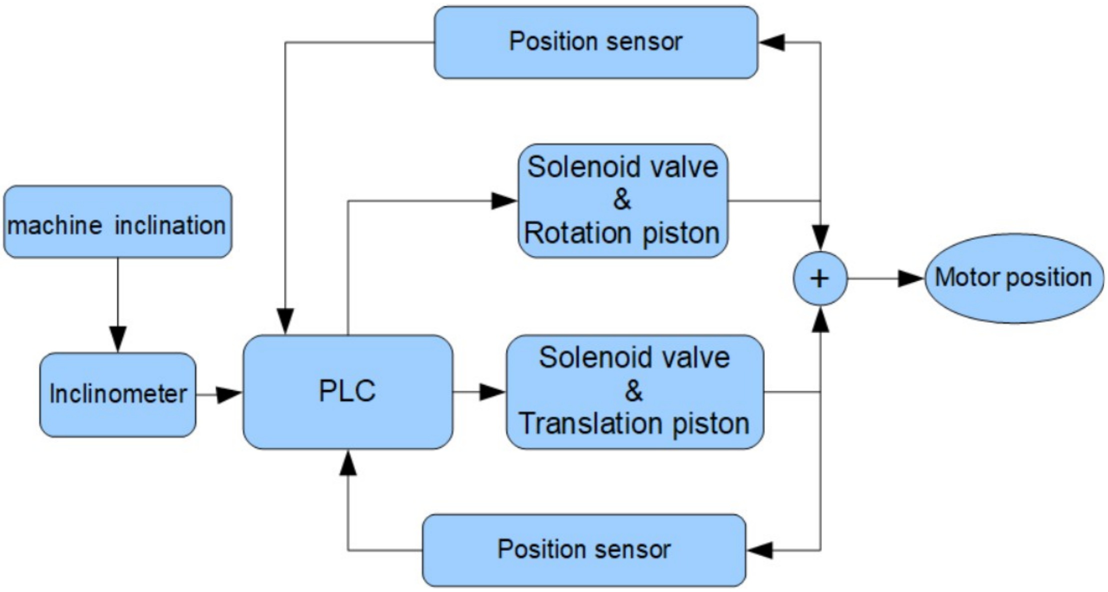


Figure 19: block scheme of the machine

Chapter 4

4.1 Sensors

In this project a crucial part is developed by sensors that help measuring correctly.
In my case there are 3 main sensors:

- One inclinometer: that measures the inclination of the motor;
- Two position sensor: that measure the position of the translation and the rotation of the motor block.

In the first tests of the machine actually we used a different sensor for the inclination, but we encountered some problems.

4.1.1 Motor inclination: first attempts

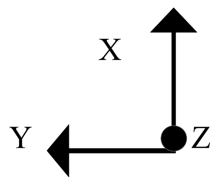
The main aspect to control in this machine is its inclination with respect to the ground, to do so we needed some sensors able to measure it.

Figure 20: IFM CR3146 Modem

(from: https://www.ifm.com/it/it/product/CR3146?gclid=Cj0KCQjw1OmoBhDXARIsAAAYGSExZdQ_JG-Ps787VifgNQT6NPIBobYZ2rOTblS3E1Oes0np-e-eomAaAk0oEALw_wcB)[4]

On the machine there is a modem, the IFM CR3146. This is mainly used as a radio interface that is devoted to connect the machine to the remote radio controller. Inside the modem there are also a gyroscope and an accelerometer. The values measured by the sensors are transmitted to the PLC through CAN bus, then the true data are extracted.
The modem is mounted on the machine exactly like in the Figure 20, with the connector pointing downwards.
From the data sheet I obtain the direction of the axis. The x axis is pointing upward, the y axis

is pointing to the left, and the z axis is pointing to the reader. An example of the axis is reported below.



As the accelerometer starts to read the data, on the x it measures 1g. On the other axis we have small values that can be supposed zero. The sensor has the value on the x axis different from zero because the accelerometer measures the effect of the gravity force [g]. As the machine moves the values start to change, then to detect the right inclination I just need to read the measured value.
Then by using the formula:

$$\alpha = atan\left(\frac{ACC_X}{\sqrt{(ACC_Y^2 + ACC_z^2)}}\right) \quad (21)$$

the inclination angle, on the x axis, can be detect.
This method uses only the accelerometer. It is not the best one because vibration generated by the motor creates a huge noise that can't be easily filtered, so we decided to use also the gyroscope.
The gyroscope is also embedded in the modem, and it measures the rate of change in [°/s] of the modem's angle. To get the right angle covered by the rotation we have to multiply the rate for the execution time of one cycle, in our case 10ms; then we obtain the angle. This value is the rotation that the machine did in the single cycle; to get the total angle we have to sum all the other angles covered during the execution.
To be more clear I make an example:

$$\omega_x = 10\,°/s$$
$$t_{cycle} = 10\text{ms}$$
$$\alpha_{cycle} = \omega_x * t_{cycle} = 10\,°/s * 10\text{ms} = 0{,}1\,°$$
$$\alpha = \sum_{i=1}^{t} \alpha_{cycle}(i)$$

then to have the correct measurement I have to use another formula that mixes up the data from accelerometer and gyroscope. This formula is called the Complementary Filter :

$$\alpha = 0.96 * \alpha_{gyro} + 0.04 * \alpha_{acc}$$

Also in this case, the noise generated from the machine was so high that it couldn't be filtered. So, to solve this problem, we decide to change sensor and we moved to a sensor that returns directly the angle without any calculation. These type of sensor are very expansive but very accurate.

4.1.2 Kalman filter

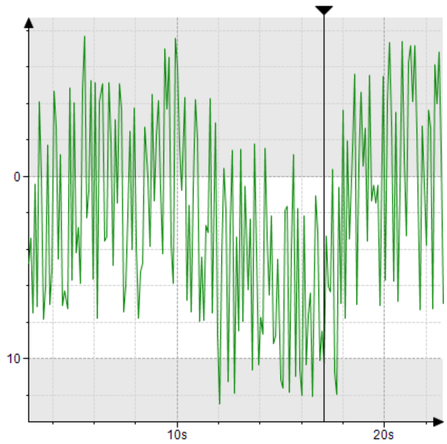
A possible solution to have the right inclination using only the accelerometer and the gyroscope, can be obtained using the Kalman filters.
The Kalman filter is an algorithm used for filtering data. It uses all the stochastic properties of the data received and by combining them, it returns a value which is the filtered form of the input, which is corrupted by a noise.
The measures of the gyroscope and of the accelerometer are corrupted by a measurement noise, which can be considered Gaussian with 0 mean value and a variance v1. The only measure that interest us, in this case, is the measure of the gyroscope. It measures the rate of change of the angle in the time, so it measures the angular velocity of the machine. The noise is mainly generated by the electronics inside the sensor and by the vibrations of the motor.
To work with Kalman filters, the system, has to be represented using state space equations..
The algorithm I used, is an adaptation of the algorithm I found on the internet at this address "http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/"[9] on the tkj electronic site.
In this algorithm, the states (the variables of interest) are two: the angle and the bias of the angular speed. The state is represented with the letter x. The output is obviously the angle, and is associated to the letter y. The external input u is the value read by the sensor.

$$x(t+1)=A*x(t)+B*u(t)+v_1(t)$$
$$y(t)=C*x(t)+v2(t)$$

The algorithm can be divided into two parts: the prediction and the filtering.
In the prediction part, the objective is to have a prediction of what will be the state at the next time instant "t+1" using all the data available at time "t". These data are: the previous value of the state and the new input u. In this part, is also calculated the new prediction variance matrix P for the predictor of the state.
Then there is the filtering part. Until now, we have the prediction of a possible future value of the state and then multiplying it by C we have a prediction for the output(the angle)
The difference between the predicted angle and the actual y(t) is called the innovation, which represent how much we are far from the actual model. If the innovation is low, we are approximating in a good way the model, if it is high the approximation is bad. For the innovation, it is also calculated its variance matrix. With this matrix and the matrix of the variance of the predicted state P is calculated the filter gain K0. This value is used to calculate the filtered value of the state and correct the value of the prediction. Then is calculated the filtered output.

$$X_f(T)=X_{pred}(T)+K_0(T)*e(T) \quad where$$
$$e(T)=C*X_{pred}(T)-Y(T)$$
$$K_0(T)=P(T)*C*(C*P(T)*C^T+V2)^{-1}$$

Above are reported the main formulas for the algorithm. All the variances are calculated at every step, since the new data provide also new information about the model. The P variance is calculated solving the Difference Ricatti Equation, and again its value is calculated at every time step.
At the end of the algorithm the variance for the predicted state is updated using the new information coming form the new data.
The results obtained using this technique are quite interesting. Unfortunately in this case, I didn't have the possibility to test it directly on the machine, since it was disassembled for the painting operation. The test with codesys anyway provided good results.
During the operation with the machine, we measured a noise with around 8-10°/s of maximum amplitude and zero mean value. It was mainly generated by the oscillations of the machine created by the motor. I tried to replicate it with a function in codesys that generates noises.
The value of the angle measured, stay still in all condition, in the figure below is reported a simulation.



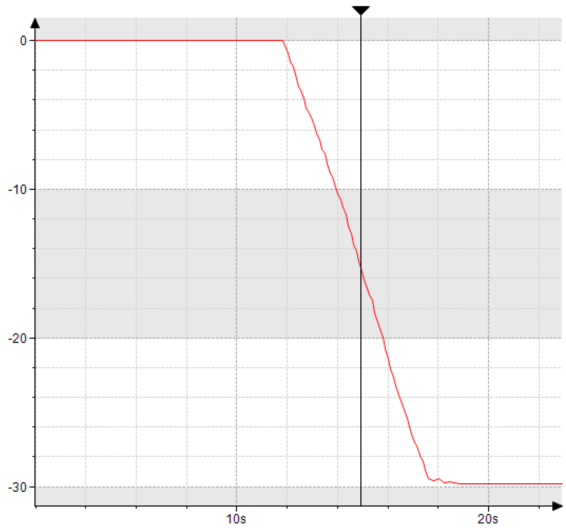


Figure 21: simulation of the kalman filter

In the picture, the green line represent a possible measure coming from the gyroscope. The red, is the angle measured. As it can be seen the measure is very noisy but the angle is almost clear. It has some fluctuation of the value, is not perfectly fixed along the time but they are small and for this project they can be ignored.

4.1.3 Sensor's selection

To select the right sensor we have to take into account some different factors.
First of all the accuracy: for sure we want something that is the closest to perfection, but since we are working on rough terrains that have lot of disconnection, we can accept some error on the degrees measure, and we decided to have at most an error of 1 degree.
Another factor to take into account is the price, as the sensor is more precise the cost will rise.

The third factor is the protection against external factors that is represented by the IP grade. The IEC 60529 standard provides a code for every grade of protection against the solid and liquid external agents. The IP grade is composed by two numbers, here is reported a Table that resumes all the codes and all the protection:

NUMBER	SOLID: first digit	LIQUID: second digit
1	Protect from object greater than 50 mm	Protection from drops
2	Object greater than 12.5mm	Protection from drops @15°
3	Object greater than 2.5mm	Drops @60°
4	Object greater than 1mm	Protect from water splash
5	Dust protection	Protect from water jets
6	No ingress for dust	Protect from water jets and flooding
7		Protect against submersion at maximum 1 m
8		Protect against submersion at 3 m
9		Protect against powerful jets at high temperature

Table 1: IP grade resume

Since the machine is supposed to work outside, even if the components are under a kind of hood, water protection is required to have the security that the electronic will work in every condition. So we compared different sensors that may work in the range of inclinations we need, and these are:

	Tsm TLP 300	Ifm JN2220	Ifm JD1111
Price in €	120	324	597
Accuracy	±0,3°	±0,5°	±0,5°
Measurement axis	3	2	1
Temperature range	-40°C...+85°C	-40°C...+85°C	-40°C...+85°C
IP	IP 67	IP 69K	IP 69K
Range measurement	±60° or ±180°	±180°	0...360
Vibration compensated	Yes	Yes	Yes
Type of sensor	MEMS, capacitive	MEMS, capacitive	MEMS, capacitive
Type of output	Analog current or voltages	Analog current	Digital value
CAN	Yes	No	Yes

Table 2: sensor comparation

4.1.4 The selected sensor

The selected sensor is the TLP 300, since it has the same characteristics of the others and it is the cheapest.

Figure 22: the TLP300 sensor

(from: https://www.tsmsensors.com/it/sensori/inclinometri/tlp300-canopen/)[10]

Another important characteristic of this sensor is that it is especially studied for agricultural purposes, so it fits perfectly in this project.

The output, as described in the Table 2, is analog, it exits a current in the range of 4 to 20 mA and then on the data sheet we can see that the minimum current corresponds to minimum angle and the maximum current corresponds to maximum measure.

This sensor has different output ranges:

- ±180°: for single axis;
- 0...360° for single axis;
- ±60° for dual axis.

On the single sensor anyway it is not possible to implement all these 3 possibilities simultaneously, so we see that the best choice for our purposes was the sensor with dual axis and ±60° range of measurements.

In the pictures below are reported some graphs present on the datasheet:

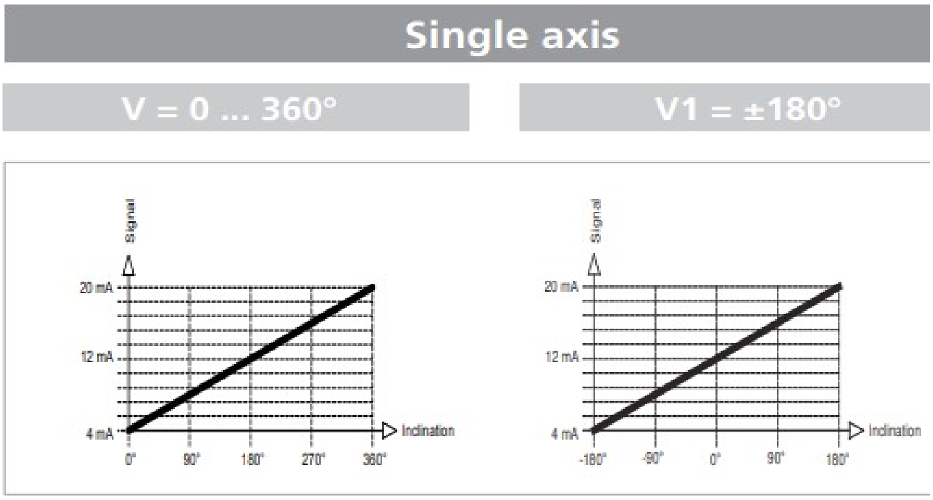


Figure 23: single axis output current, as can be seen for the version 'V' the output measurement is from 0° to 360°, while in the 'V1' the range is ±180°

(from: tlp300 datasheet)[10]

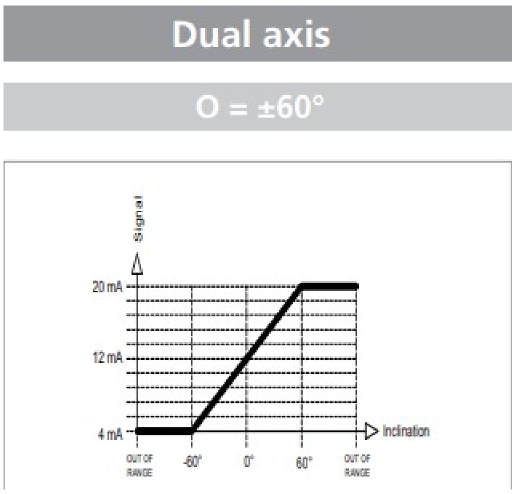


Figure 24: here is represented the output of the sensor in the case of the dual axis mode, the range of measurement is shrinked to ±60°

(from:tlp datasheet[10])

4.2 CODESYS: the programming environment

Codesys is a German company that develops software for automation.
In this thesis we used the software called CODESYS V3.5 SP 11 that was proposed in the IFM site when the PLC was chosen.
CODESYS is a development software where the code for the PLC is written. The language is an object oriented language, whose syntax is similar to all others programming languages, the user can create its own functions or programs and can use them in lots of ways.
The program is very simple to use, it provides lots of functions that can also be extended by installing new libraries in the system. It also provides an inner compiler that can help during the simulation of the system to detect errors.
One of the most important libraries in the system is the OSCAT basic. It provides all the math functions that can be needed in a possible program. For example, it provides the trigonometric functions, functions for filtering signals or functions to create signals when needed.
This program also provides a graphical tool that lets the user visualize the signal on screen.
As last thing the program is very widely used in the automation sector, so it provides the possibility to help the user to add easily devices on the network.
In order to let the sensor communicate with the PLC I had to insert the sensor in the network and CODESYS helps us.[11]
CODESYS provides a list of devices to the user that can be easily connected to the Can bus and then to the PLC. In this case the TLP300 wasn't in the list of possible devices because the list contains only a small part of what can be really connected to the PLC. So we have to add it manually.
To do so first of all I had to install the 'tlp300.eds' file in the list. The '.eds' extension means 'Electronic Data Sheet', then inside this files are described all the characteristics of the device we are using and which parameters can be changed or modified.

```
[FileInfo]
CreatedBy=TSM Sensors
ModifiedBy=TSM Sensors
Description=TLP300-H-FW2102R0105-Canopen
CreationTime=02:12PM
CreationDate=04-17-2023
ModificationTime=02:12PM
ModificationDate=04-17-2023
FileName=EDS0007 R00 TLP300-H-FW2102R0105-CANopen.eds
FileVersion=0
FileRevision=0
EDSVersion=4.0
```

Figure 25: example of information inside an eds file

Another information that can be found in the eds is for example the information about the COB-ID.

```
[1800sub1]
ParameterName=COB-ID
ObjectType=0x7
DataType=0x0007
AccessType=rw
DefaultValue=$NodeID+0x00000180
PDOMapping=0
```

Figure 26: information about COB-ID

Once the device is installed, the communication starts and the data are transmitted to the PLC. For this part of the project I am interested only in the rotation around the x axis. Data arrives in integer form so, since the resolution of the sensor is 0,01°, an angle of 12.65° will arrive in the form of number 1265. So to read the sensor correctly I had to divide the number received on the CAN bus by 100.

As expected when the machine starts, the oscillations generate by the motor don't affect the measurement which is still and clear every time.

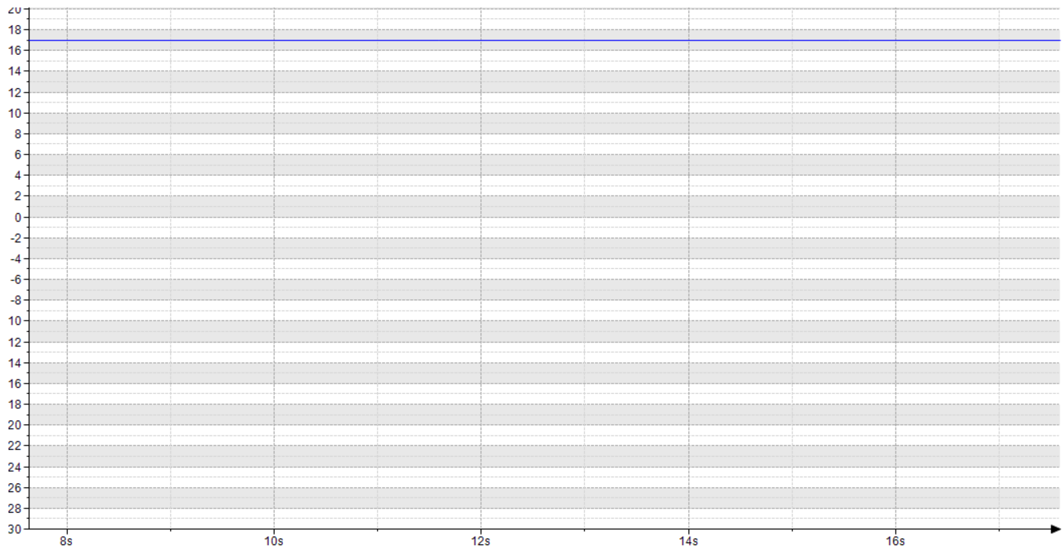


Figure 27: sensor measurement, the blue line represents the measure of the inclination of the machine along the time

As we can see from the picture, the machine inclination is 17° and it stays still along the time, even if the machine is on and the motor produces vibrations.

4.3 MEMS and capacitive sensors

Most of the sensors nowadays are based on the MEMS technology. MEMS is acronym that means Micro Electro Mechanical System and they are used in a lot of applications thanks to their characteristics. In this thesis MEMS are used in the sensors for inclination. MEMS are small electronic devices that embeds some mechanical system and that are used a lot because they are small and consume a small amount of power.

4.3.1 Capacitive sensors

The tlp 300 is a capacitive sensor. This means that there is a capacitance inside it that changes its value when the inclination changes.
Each time we have a layer of dielectric material between two metallic, a capacitance is formed. The formula for a capacitance is the following:

$$C = \epsilon_0 * \frac{A}{D}$$

Where A is the area, D is the distance and ε_0 is the dielectric constant and it is equal to $8.85\,x10^{-12}\frac{N^2}{C*m^2}$.

The basic principle can be represented with a mass connected to some spinrg like in the figure 28.

Figure 28: basic model of a capacitive accelerometer

By using this model we have three components:

- fixed plates, that are a reference for the "still" capacitance;
- mass that is free to move as the sensors moves;
- springs that connects the mass to the plates.

$$C_1=\epsilon_0*\frac{A}{xo+d}$$
$$C_2=\epsilon_0*\frac{A}{xo-d}$$

With this model we have two capacitance in parallel. Since now it can't be measured anything. If we connect the two plates to the same sinusoidal voltage, but to one the voltage is inverted, it is created a kind of voltage divider. Then if we measure the voltage between the mass and one plate, we can understand the movement of the sensor.

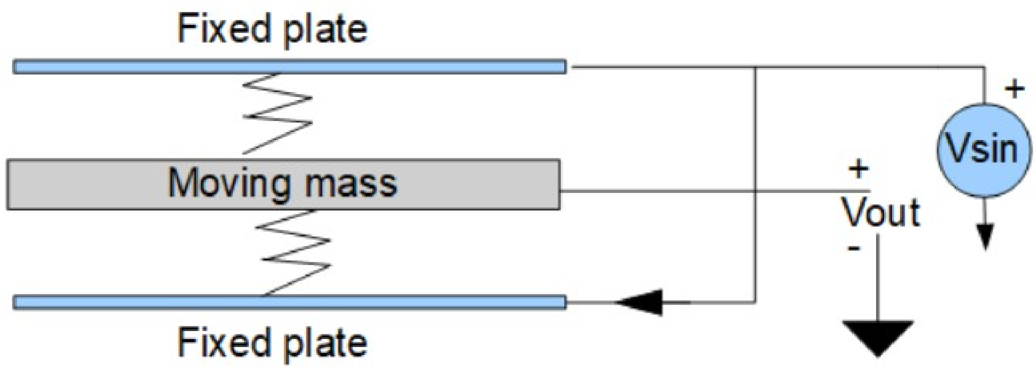


Figure 29: measurement circuit for displacement sensor

Then to calculate the acceleration we can use this system.
The mass is connected to a spring. The spring is governed by the equation $F=K*\Delta x$ this force is the same acting on the mass. Using the second law of dynamics the force is $F=m*a$, so to know the acceleration, the formula is:

$$a=\frac{k*\Delta x}{m}$$

Since the displacement of the mass is small, the value measured is small.
To increase the quantity measured, it should be increased the value of the capacitance that changes.
In order to get the maximum capacitance variation we should have the largest possible area of the plates or the minimum distance between them. To achieve this goal inside the sensor are built some kind of ramifications, so that with this technique the area of the plates is increased.
If we want to have a measurement, only the area is not enough. We are looking at something that changes with movement, so we need that the capacitance of the sensor changes too. To do so the plates of the capacitance are attached to some springs that let the system move when the sensor moves.[12]

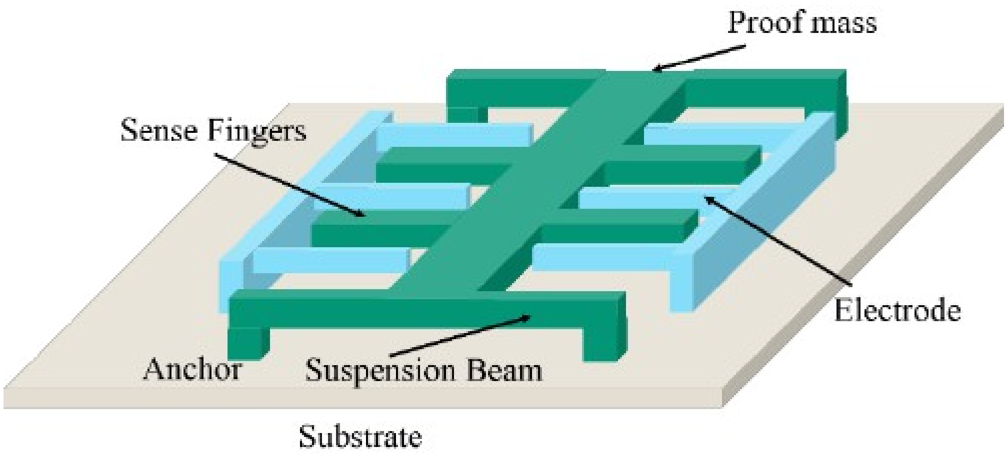


Figure 30: simple scheme of an accelerometer

(from *Design and Simulation of MEMS Differential Capacitive Accelerometer,(2014 ,S.Sinha)[12])*

Then to measure the acceleration we have to measure the variation of the capacitances. To do so first of all a sinusoidal voltage is needed and it is connected to the fixed electrode; then, as the mass moves, between the plates is generated a voltage that once measured and amplified can represent the acceleration measured by the sensor.

Another technology that can be used to build accelerometers or vibration sensors, is by using the piezoelectric effect.

The basic principle is the same, a model of the piezoelectric sensor is reported in the next figure

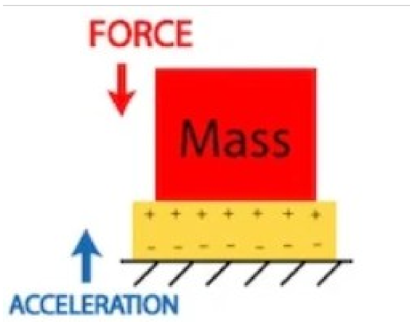


Figure 31: model of a piezoelectric sensor

*from(*Understanding Piezoelectric Accelerometer BasicsMarch 20, 2022 by *Dr. Steve Arar[13]*)

There is a mass free to move or vibrate, but then the acceleration is calculated by measuring the voltage generated by the piezoelectric material. As it feels the vibrations or a compression the piezoelectric material will generate a voltage proportional to the acceleration that it feels, then since this value is very small it is added an amplifier that increases the voltage to a reasonable level so it can be measured..[14]

Chapter 5

5.1 Weight management

This is the crucial part of the project.
At this point, there is a machine that can move on the terrain. If it encounters any hill or hole and the inclination is no more zero, it has to move the motor and balance the difference of weight distributed on the tracks.
In order to do so, the company mounted two pistons between the chassis and the motor block. These two will extend or shorten in order to have the right position of the motor.
For doing so we needed :

- Two pistons;
- Two solenoid valves;
- Two position sensors.

The two pistons are dedicated one to the translation and one to the rotation.
As they increase or decrease their length, the pistons change the position of the motor. The same thing happens when the rotation piston moves. The motor, then, can translate and rotate. Once the machine moves and finds a not planar surface, the process of self balancing starts immediately. In order to know the position of the motor, again, we had to choose among lot of possibilities and technologies, and we decided to use the position sensor IFM OMH551.

5.1.1 Ifm OMH551

This sensor is a photoelectric sensor so it measure the distance using light.
Photoelectric sensors use the light to measure. They convert this light into an electrical signal that can be used by the other electronics.

Figure 32: the omh 551 sensor

(from:https://www.ifm.com/it/it/product/OMH551)[4]

Photoelectric sensors are very precise and have the advantage of measuring without contact, and thanks to the power of light they can arrive to measure large distances.
The principle at the base of these sensors is very simple. Basically they are composed of an emitting source that emits a light beam, this beam will collide with the detected object then it will be reflected. At this point, the reflected beam will reach the receiver which is a photo transistor that generates a current based on the light that hits it.[15]

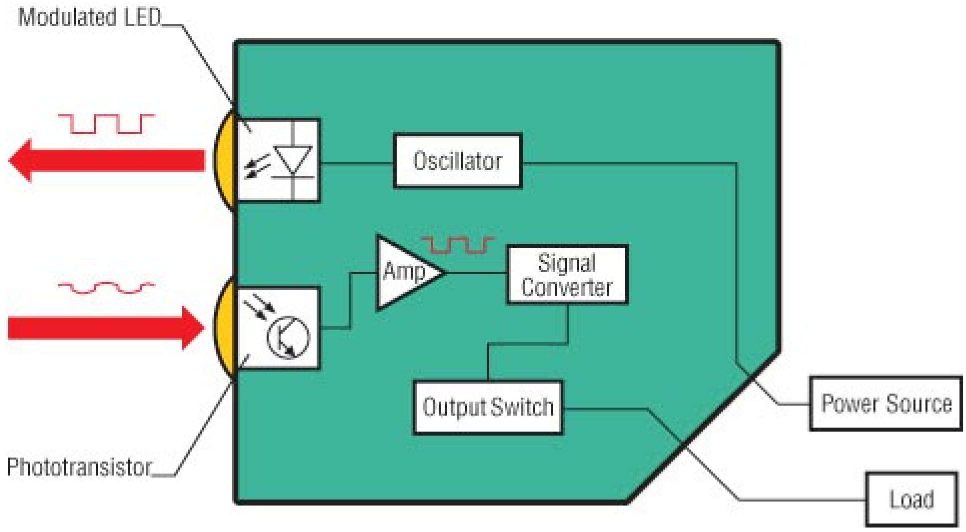


Figure 33: model of a photoelectric sensor

(from: Gary Frigyes, Ed Myers and Jeff Allison *, fundametal of fotelctric sensors,(2013), www. Automation.com)[15]*

In this case I needed precision since distance are quite small: in fact we had to measure displacement that ranges between 3,5 and 8 cm and the sensor range perfectly fits in this interval.
Another important aspect of this sensor is that it has the IP67 grade of protection, this is fundamental because it has to work in very hard conditions, in dusty or wet environments.
This sensor returns a current as output that is in the range of 4mA to 20mA, so if the sensor measures 3 cm the output will be 4 mA, then this current will be measured by the PLC and converted into a distance following a linear characteristic.

5.1.2 Sensors in the machine

Position sensors are required in order to calculate the position of the motor on the chassis.
In order to have a correct measurement, the sensors are placed in a manner where they have to measure a small space so there is low probability of having object interfering with it.
In the picture below it can be seen the chassis of the machine:

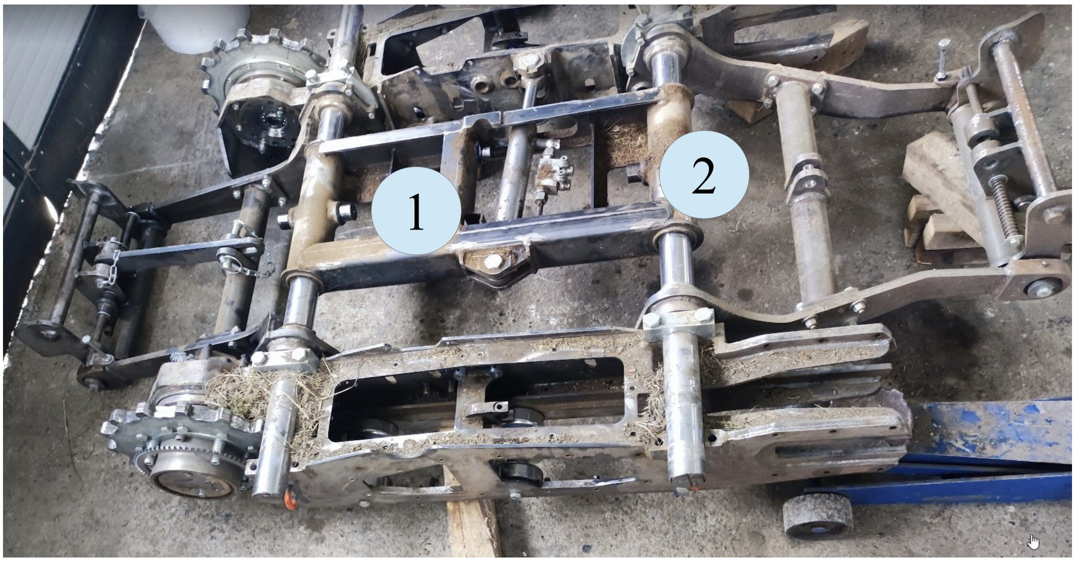


Figure 34: machine's chassis without tracks and motor

The two displacement sensors are positioned in the position 1, for the one that controls translation, and 2 for the sensor that controls the rotation.
In the Figure 34 we can also see the piston that is used to translate the motor. As it can be seen there are two tubular guides that support the motor. Onto these guides the motor also moves when the piston is actuated.

5.1.3 solenoid valves

Solenoid valves are the interface between the code written and the pistons.
They receive a signal from the PLC and then they open/close the oil flow to the piston. The selected valves are the Duplomatic BDL1.

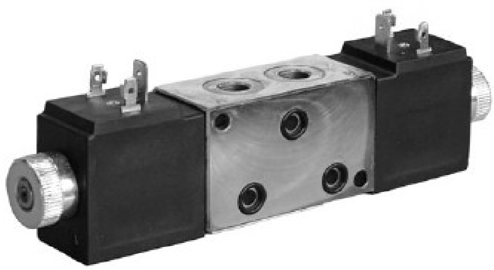


Figure 35: BDL1 valve

from (https://duplomaticmotionsolutions.com/ [16])

The working principle is quite simple, there is an electronic circuit made with a solenoid. Depending on if that solenoid is excited or not, the valve opens or close.

5.2 Motor position

To know the exact position of the motor I had to read the data from the two sensors.
The sensors return a current value in mA then, since the characteristic of the sensor is linear, the conversion can be easily done.
As it can be seen in the Figure 26 there are two horizontal guides over which the motor translate; between these guides there is a support where the sensor for the translation is placed. In the picture 36 is represented a scheme of the system that translates the motor.

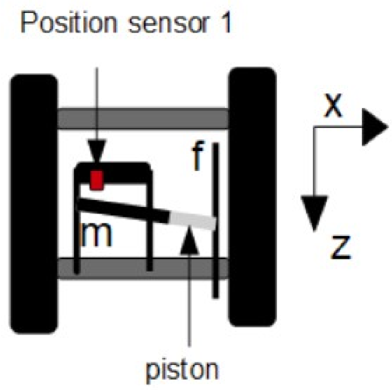


Figure 36: translation system scheme

As it can be seen, the piston is connected between two plates, one mobile 'm' and one fixed 'f'. Then there is a support for the sensor that translates with the motor. As the piston elongates, the sensor measures a different distance as shown in figure 37.
The big problem of this procedure is only in the non linearity of the conversion from the measure we obtain and the real position of the motor. On the boundaries of the measures there wasn't any problem and the measure was correct but then, if the motor is in the central position the distance to be measured is expected to be the half of the range, but it is not like that.
The relationship between the sensor measure and the motor position can be represented in figure 38.
In figure 37, is represented instead what happens when the motor moves. As can be seen, as the piston elongates (2) so also the sensor changes its position. It can also be seen another thing: as the piston elongates, the distance to be measured (from sensor to piston) changes and in particular it reduces and the relationship between the elongation and the measure of the sensors is not linear. So with the company we decided to move from a singular function to a linear piece wise function. Thanks to a function already present in CODESYS we had the right function that approximates the measurement and the position of the motor.

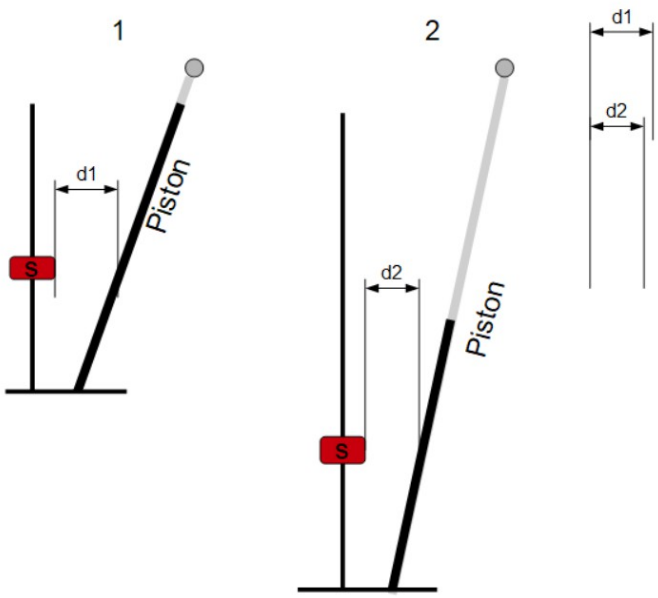


Figure 37: how the position of sensor and the piston length change

The function used is called "Char Curve" that takes in input a sequence of points and then if we feed to the function a new point it will return us the expected output we need.
To do so we measured by hand some points. These point are made of two components, which I will call x and y for simplicity.
Each couple (x,y) was measured in this way:

- x coordinate: is measured with a measuring tape and it's the distance between the fixed plate 'f' and the mobile plate 'm'
- y coordinate: measure got from the sensor.

The result obtained is this

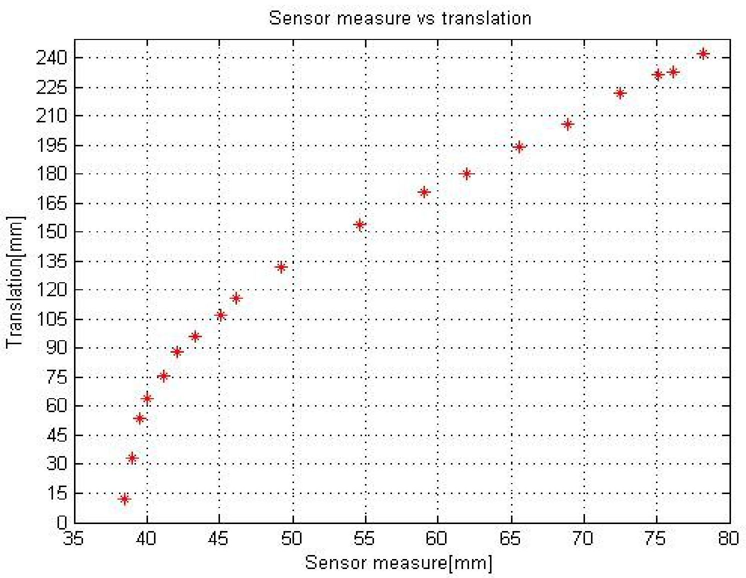


Figure 38: measurement and translation relationship

As can be seen the relationship is strongly not linear when the measure is small, so when the piston has maximum length.
Then this set of points is inserted as input in the function. From now on if the 'Char Curve' function will receive new data, so a new measure from the sensor, it will return a value that is consistent with this curve.
For what regards the inclination, the principle is nearly the same. There is a piston moving the motor, but now is placed between the tubular guide and the motor. In this way if the piston elongates, the motor rotate, if the piston shorten the motor will rotate in the opposite way. In order to know how much rotation is done, the second position sensor is positioned on the motor. On the rotation axis of the motor, there is a piece of metal that is not perfectly circular. In this way as the motor, and with it the sensor, moves, the sensor will measure a different distance. In this case the measure was linear, so at 4 mA we have the maximum rotation in one direction, at 20 mA we have maximum rotation in the other sense. If the motor is in the central position the measure given by the sensor is around 12 mA so, half of the range.
In the following picture is represented the position of the sensor in the machine

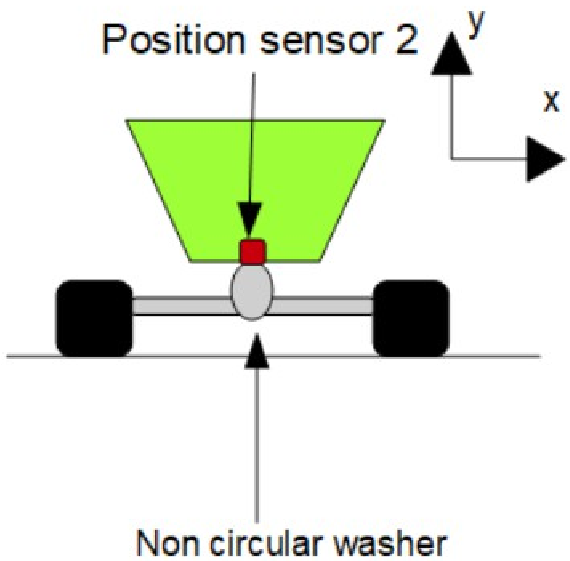


Figure 39: placement of the position sensor for the rotation

Now, the machine, as it feels a variance in the slope of the terrain can rotate and translate.
The movement will start as the machine sense an inclination greater than 1.5°. If the inclinometer senses a lower value, the rotation will not start.
The machine has two possible solution for doing these movements.
One of them is that the user drives directly the two pistons with the radio command. In this case none of the control on the position of the motor is active, the motor is free to move as the user wants and has the only restriction of remaining inside the limit of translation (+ 105 mm or -105 mm with respect to the center of the guides) and rotation(+25° or -25° starting from the vertical position).

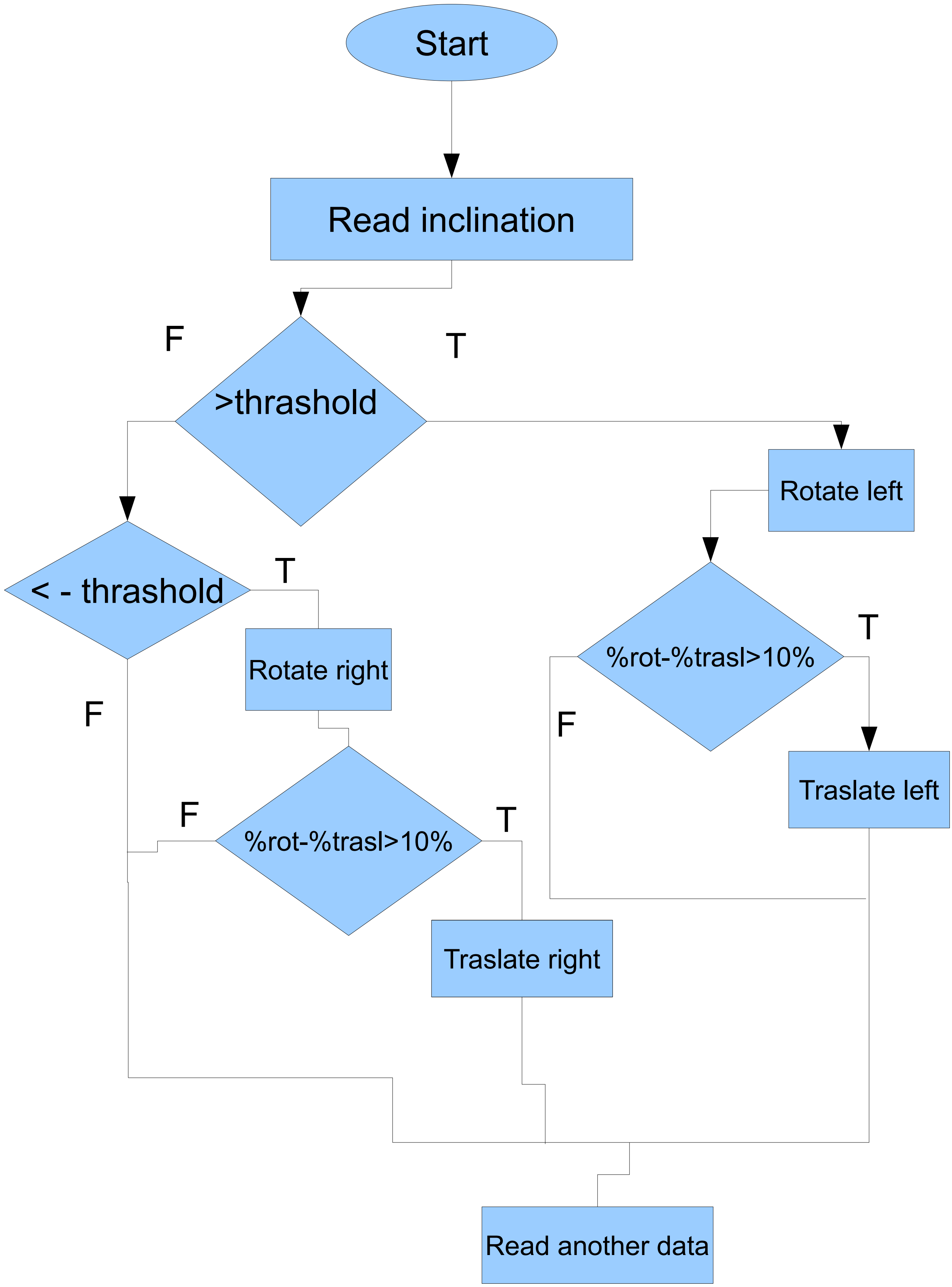
The second possibility is to have an automatic procedure that controls the movements. The sequence of action is the following:

1. measure of the inclination of the motor;
2. compare with a threshold;
3. if bigger rotate left or right depending if positive or negative
4. compare the percentage of the already done rotation and translation;
5. if rotation is 10% bigger than translation also translate;
6. activate the output signals towards the valves.

To be more clear I report in the next page a flow chart.

As it can be seen before doing the movement the system checks if the inclination is positive or negative. In the case in which it is positive, the motor has to rotate and translate to the left. In the opposite case, so with negative slopes, the movement is towards right.

A better explanation will be done in the next paragraph, where I will explain the crucial parts of the code.


Start
Read inclination
F
T
>thrashold
Rotate left
< - thrashold
T
T
%rot-%trasl>10%
Rotate right
F
F
Traslate left
F
T
%rot-%trasl>10%
Traslate right
Read another data

5.3 CODE DESCRIPTION

The code, as said before was written using CODESYS.

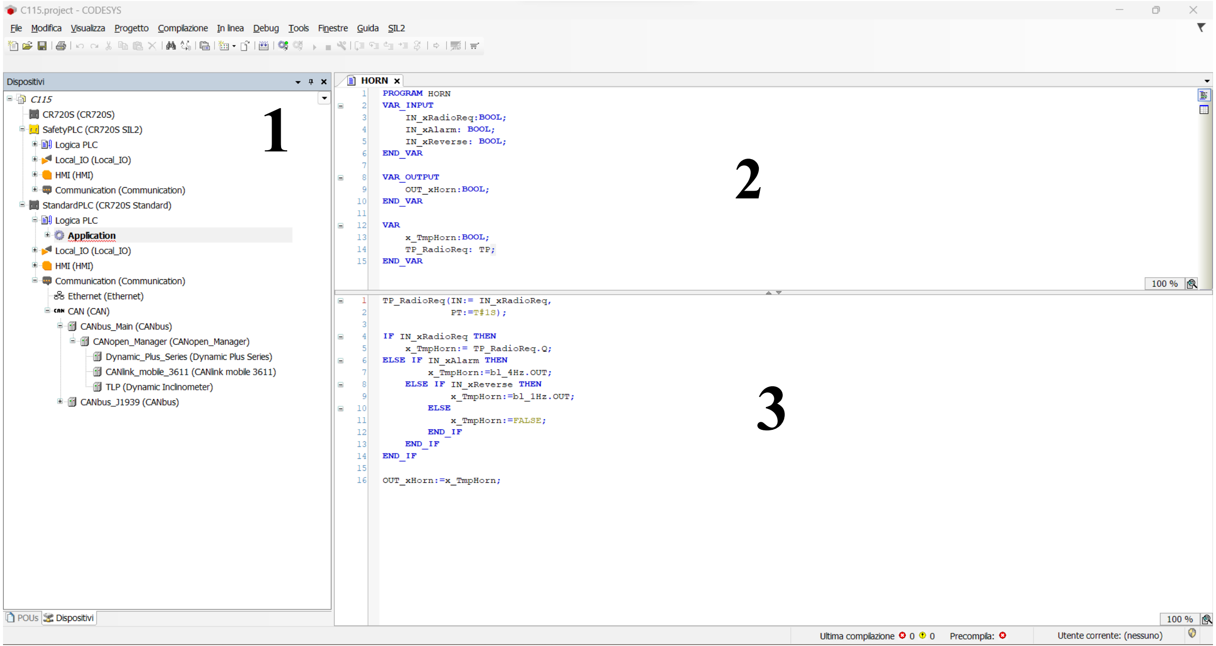


Figure 40: Codesys interface

In the previous picture is represented, the first screen of CODESYS. The page can be divided into 3 parts:

1. The list of objects present in the network. It specifies for which purpose they are dedicated, in particular if are PLC/ Safety PLC or are dedicate to communication or if are present some graphical interfaces.
2. Is the part dedicated to the variables. CODESYS divides them in three categories: Input variables, Output variables, general variables. Input variables obviously are the inputs of the program, output variables are the outputs.
3. The part where the code can be written

Each program of the project can be divide into three smaller parts:

- read the input from sensors/ CAN bus;
- write the logic for the functioning of the machine;
- associate the output pins to some variables in the code.

```
PROGRAM HORN
VAR_INPUT
     IN_xRadioReq:BOOL;
     IN_xAlarm: BOOL;
     IN_xReverse: BOOL;

END_VAR

VAR_OUTPUT
     OUT_xHorn:BOOL;
END_VAR

VAR
     x_TmpHorn:BOOL;
     TP_RadioReq: TP;
END_VAR
```

Figure 41: example of variables division

By doing this separation, the risk of overwriting some variables and then having some problem, in the code is reduced.
Another important thing is to consider all possible scenarios. For example in a “if” statement is mandatory to have always an action both for the “true” or “false” case. In this way the code is always in a part written by the author and the machine didn't go in states where actions aren't defined.

5.3.1How to program with codesys

As said before, there are different ways to program a PLC. In this project we used both Structured Text (ST) and Function Block Diagram(FBD).
ST is similar to a classical programming language, where the user writes directly the statements he wants to do, this is useful to describe actions that a part of the code has to do.

```
TP_RadioReq(IN:= IN_xRadioReq,
            PT:=T#1S);

IF IN_xRadioReq THEN
    x_TmpHorn:= TP_RadioReq.Q;
ELSE IF IN_xAlarm THEN
        x_TmpHorn:=bl_4Hz.OUT;
    ELSE IF IN_xReverse THEN
            x_TmpHorn:=bl_1Hz.OUT;
        ELSE
            x_TmpHorn:=FALSE;
        END_IF
    END_IF
END_IF

OUT_xHorn:=x_TmpHorn;
```

Figure 42: example of code in ST

In FBD mode, CODESYS, provide to the user some blocks that put together they can create a program. This is very useful when functions need to be called, because it shows clearly which are the inputs and the outputs of the functions.

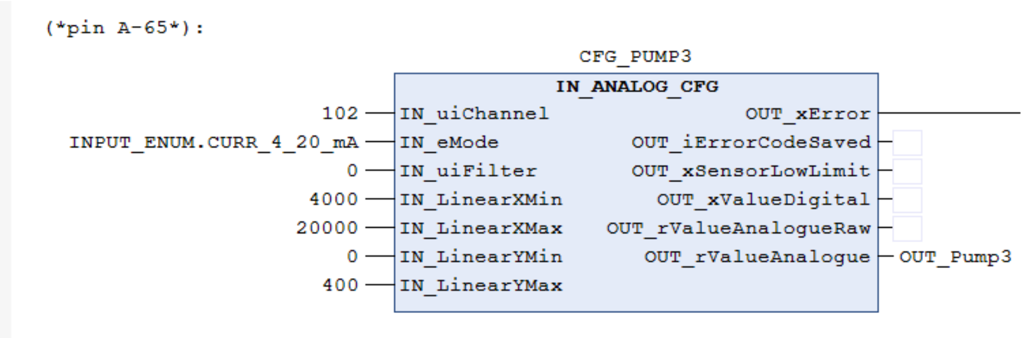


Figure 43: example of a FBD block

As it can be seen, using this type of programming mode, all the inputs and outputs are separated. All the action that this block does are described in the FB IN_ANALOG_CFG.
When is needed a new piece of code we can decide if it can be a :

- PRG: Program, a classical program with inputs and outputs, that is executed every time. To instantiate it is necessary to call it directly, it can't be used as a structured variable.
- FB: Function Block, similar to a program, but is executed only if called. In a program if we want to call a function block it can be used also as a structured variable
- Function: Classical function with input and one output as a return value.

FB are useful to create some smaller program, inside the big one that do all the things needed.
All the prg or the FB that are described in the program can be utilized as a FBD block.
The main program is the PLC_PRG. It is executed continuously in a cyclic way. Inside of it are present all the actions that controller should do.
There are also other possibilities to program a PLC in CODESYS, like the ladder diagram (LD) or sequential function chart (SFC), but we didn't use them in this project.

5.3.2 Variables names

As can be seen in the previous figures, the variables have strange names. The rule for naming a variable is composed of three parts:

- Classification: if is an input we put on the front the prefix “IN_”, if it is an output “OUT_”, if is a general variable it doesn't have any prefix.
- Type of variable: after the prefix is put an indication (the first letter usually) for indicating the type of variable. For example if a variable is a real and an input, the prefix become “IN_r_”.The main type with their abbreviation are INT=i, REAL=r, WORD=w. A bit different rule is used for the BOOL type, in this case is used the letter x.
- Real name of the variable: it should be the most descriptive possible, in order to easily

remember which is the main goal of that variable.
For example in the figure 42 the variable called "OUT_xHorn" will represent the output of that block. In particular it is a Boolean variable which is dedicated to activate the horn of the machine.

5.3.3 Angle calculation

The program I Wrote is a part of a larger work, so it is a small part. In this type of project,as said before the main program is the PLC_PRG, that is the only one continuously executed. Inside the PLC_PRG are present all the calls to other PRG or FB. If one object isn't called in the main program, it will not be executed.

```
ANGLE_CALC( IN_xEnable := TRUE,
            IN_xInit:= xInit,
            IN_rRollRaw := INT_TO_REAL(INCL_X),
            IN_rPitchRaw := INT_TO_REAL(INCL_Y),
            IN_rRelativeRoll := rBodyIncl_deg,
            IN_rRelativePitch := 0.0,
            OUT_rMachineRoll => rRollMachine,
            OUT_rMachinePitch => rPitchMachine,
            OUT_rTerrainRoll => rTerrainRoll,
            OUT_rTerrainPitch => rTerrainPitch);
```

Figure 44: Call of a PRG in the PLC_PRG

As it can be seen, here the block "ANGLE_CALC" which is the one where originally were made the calculations with accelerometer and gyroscope. Now is used simply to read the data from the inclinometer.
The real data from the inclinometer, come from the CAN bus, then is saved in the two variables, INCL_X and INCL_Y.
At the beginning of each block, or at least of the most important, are present two boolean variables:

- IN_xEnable: which is the enabling of the block, in this case is always set to TRUE since this block is always enabled. If the enable is set to false the block will be executed, but the outputs will be set to a default value.
- IN_xInit: if this field is set to true, all the variables inside the block, that can't be seen now, are set to a predetermined value. This allow to initialize all the useful variables. Normally, is set to true for the first cycle, then is set to false.

If we enter in the block, it can be seen that the value we get from the can bus is measured in cents of degree, so an inclination of 11,34° is reported as 1134. In order to get the right value we have to divide it by 100.
As said before if the enable is set to false, all the outputs are set to 0.
another thing can be seen. All the outputs are assigned only one time and at the end of the code for avoiding to overwrite them.
The inclinometer, for mechanical reasons, was mounted in a sense opposite to the logical one,

so to simplify the calculations the value are inverted of sign.

```
IF ( NOT IN_xEnable) THEN
    OUT_rMachineRoll := 0.0;
    OUT_rMachinePitch := 0.0;
    OUT_rTerrainRoll := 0.0;
    OUT_rTerrainPitch := 0.0;
    RETURN;
END_IF

// 0.01 deg scale, due to mechanical installation the roll axis is opposite [deg]
rRoll:= (IN_rRollRaw / 100 * -1) + rRollOffset;
// 0.01 deg scale [deg]
rPitch:= (IN_rPitchRaw / 100) + rPitchOffset;

//[deg]
rTerrainRoll:= rRoll + IN_rRelativeRoll;
//[deg]
rTerrainPitch:= rPitch + IN_rRelativePitch;

//set output
OUT_rMachineRoll := rRoll;
OUT_rMachinePitch := rPitch;
OUT_rTerrainRoll := rTerrainRoll;
OUT_rTerrainPitch := rTerrainPitch;
```

Figure 45: code of the Angle Calc program

As last thing, are also calculated the pitch and roll of the terrain, these will be used for security reason. If the machine will go on a dangerous zone, with too high lateral or longitudinal slope it will send a message to the user and if the message will be ignored the machine will stop.

5.3.4 Weight manager

Another interesting part of the code is the one dedicated to the movement of the weight. It takes as input lots of parameters, like the translation and rotation limits, the Enable, the Init and then all the commands coming from the radiocontroller.

The movement of the motor can be done in two modalities: one manual,where the user through the controller controls directly the motion and one automatic, where all the movements are controlled by the PLC. The manual mode is mainly meant for helping the user during the maintenance, while during the actual work the machine is supposed to be in automatic mode.

```
xRightTranslationMaint := IN_xCmdTranslationRight;
xLeftTranslationMaint := IN_xCmdTranslationLeft;
xRightInclinationMaint := IN_xCmdInclinationRight;
xLeftInclinationMaint := IN_xCmdInclinationLeft;
```

Figure 46: input association in maintenace mode

As it can be seen in this case the “IN_xCmd...” which are the variables corresponding to the buttons of the remote controller are assigned to some intermediate variables, which will be, in

turn, assigned to the output variables.
To pass in automatic mode, the user has to press a button on its controller. This button, if pressed again will exit from the automatic mode.

```
// --- Rotation auto ---
TON_InclRight(IN:= (IN_rRollAngle_deg < -rInclinometerThreshold), PT:= T#300MS);
TON_InclLeft(IN:=  (IN_rRollAngle_deg > rInclinometerThreshold), PT:= T#300MS);

// On valve activation if error is bigger than thereshold
IF TON_InclRight.Q THEN
    xRightInclinationAuto:= TRUE;
END_IF

// Off valve when reach -1.5 or limit, there is a 1.5 deg of delay when stop the valve
IF (IN_rRollAngle_deg > -1.5) OR (IN_rBodyIncl_deg > IN_rInclinationLimit) THEN
    xRightInclinationAuto:= FALSE;
END_IF

// On valve activation if error is bigger than thereshold
IF TON_InclLeft.Q THEN
    xLeftInclinationAuto:= TRUE;
END_IF

// Off valve when reach 1.5 or limit
IF (IN_rRollAngle_deg < 1.5) OR (IN_rBodyIncl_deg < -IN_rInclinationLimit) THEN
    xLeftInclinationAuto:= FALSE;
END_IF
```

Figure 47: code for rotation

As the automatic mode is inserted, first of all is compared the angle of inclination with the treshold (5°). If the inclination is higher the system starts to move, in particular it rotates. The rotation is activated through a TON block. These type of blocks produces a Ton delay, so if the input becomes true the output will become true after a certain time, in this case 300ms. The delay is inserted to help the valves to work correctly. In figure 47 can be seen that rotation command become false if the angle measured is >-1.5 or <1,5. We added this condition, because we have seen during tests that the machine when stops the rotation produces always an error of 1.5°. This error is generated by the delay that is present between the stop command and the actual moment where the machines stops to move.
An interesting thin that we are studying is to add a kind of predictive algorithm that on the base of the inclination, temperature and pressure of the oil decides when is the right moment to stop the movement.
The code works on the base of the difference of the percentage between the two movements. For example, if the machine is rotating and the % of rotation is 10% bigger than the %of translation, the machine stops to rotate and starts to translate. At each cycle these percentages are controlled, so the system can rotate or translate in every cycle of the machine. Again, to activate the valves we wait for 300ms to let them work correctly. (fig.47)
These movements are done until the system reaches the translation and rotation limits.
Once all these movements are done, the output of the block are set.
Since now the program worked with intermediate variables, now depending on the modality of work the PLC will assign them to the real output of the block, which correspond to the

input of the valves that control the movements. (fig.46)

```
// Set output
OUT_xAutoAlignmentEnabled := xAutoAlignmentEnable;

IF IN_xMaintenance THEN
    OUT_xTranslationRight := xRightTranslationMaint;
    OUT_xTranslationLeft := xLeftTranslationMaint;
    OUT_xInclinationRight := xRightInclinationMaint;
    OUT_xInclinationLeft := xLeftInclinationMaint;
ELSE // Automatic control
    IF xAutoAlignmentEnable AND NOT IN_xSensorError AND IN_xEngineRunStable THEN
        OUT_xTranslationRight := xRightTranslationAuto;
        OUT_xTranslationLeft := xLeftTranslationAuto;
        OUT_xInclinationRight := xRightInclinationAuto;
        OUT_xInclinationLeft := xLeftInclinationAuto;
    ELSE
        OUT_xTranslationRight := FALSE;
        OUT_xTranslationLeft := FALSE;
        OUT_xInclinationRight := FALSE;
        OUT_xInclinationLeft := FALSE;
    END_IF
END_IF
```

Figure 48: output association

```
        xLeftTranslationAuto:= TRUE;
    END_IF

    // Off valve when reach 10 or limit
    IF (rDiffPerc < 10) OR (IN_rBodyTrans_mm < -IN_rTranslationLimit) THEN
        xLeftTranslationAuto:= FALSE;
    END_IF

    // reset translation if opposite command, priority to inclination
    IF xRightInclinationAuto AND xLeftTranslationAuto THEN
        xLeftTranslationAuto := FALSE;
    END_IF
    IF xLeftInclinationAuto AND xRightTranslationAuto THEN
        xRightTranslationAuto := FALSE;
    END_IF
```

Figure 49: settings for the automatic rotation

5.3.5 Visualization

CODESYS provide also the possibility to simulate the code written by using a graphical editor. In this editor we have the possibility to create graphs for have a trace of what the sensor are measuring or the user can put some button or leds to test the boolean variables.
With the visualization tool it can also be designed the model of the machine, like the one in the picture 50.
In the figure, is represented a model of the machine, the green part represent the motor, it rotates as the scroll bar on the right moves.
The created simulation is added to the list of object in the interface.

Figure 50: simulation with codesys

Chapter 6

6.1Turn over

One problem that has come out while testing the machine was: "what is the limit over which we can't go otherwise we have the turn over of the machine?". Obviously, since this machine is the first that the company produced, we had never tried this limit but, we make some calculation to find it out.

First of all, a body or a mass will turn over when the vertical line passing through its center of mass falls outside of the support plane. In the picture below are represented the two possible cases, the green dot s the center of gravity, while the blue line is the support base.

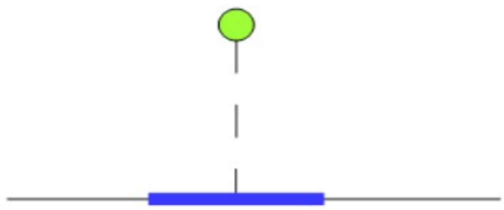


Figure 51: here the vertical crosses the support plane, we don't have any overturn

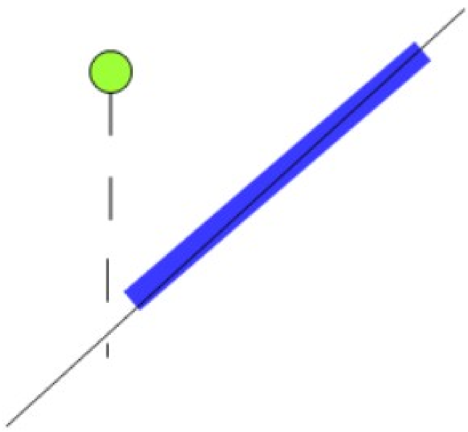


Figure 52: here the vertical line falls outside of the support plane, there will be overturn

To study better the capacity of this robot, I had to study also these characteristics.

6.1.1 Machine center of mass

Our machine is composed basically of three parts:

- The motor;
- The chassis with tracks;
- The tool on the front or on the rear.

The machine can be seen as a body composed of three masses:

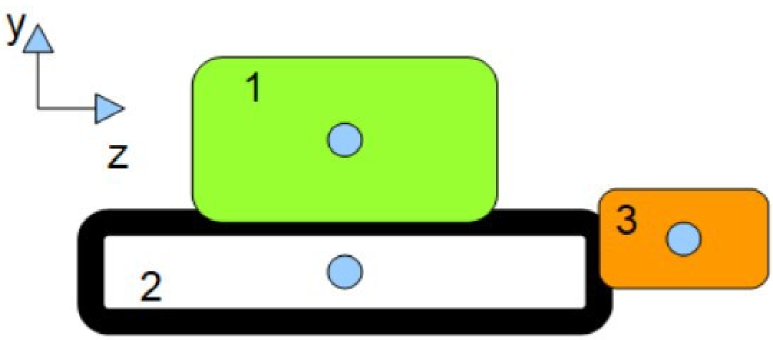


Figure 53: lateral view of the model of the machine

The green mass is the motor whose weight is 650 kg, the black one is the chassis and the tracks, this weights 500 kg, then there is the orange mass that is the tool carried by the machine,whose weight is around 250 kg.
First of all I need to set the origin of the coordinate system that I use, then I can extract the coordinate of the center of mass. The system can be represented in this way:

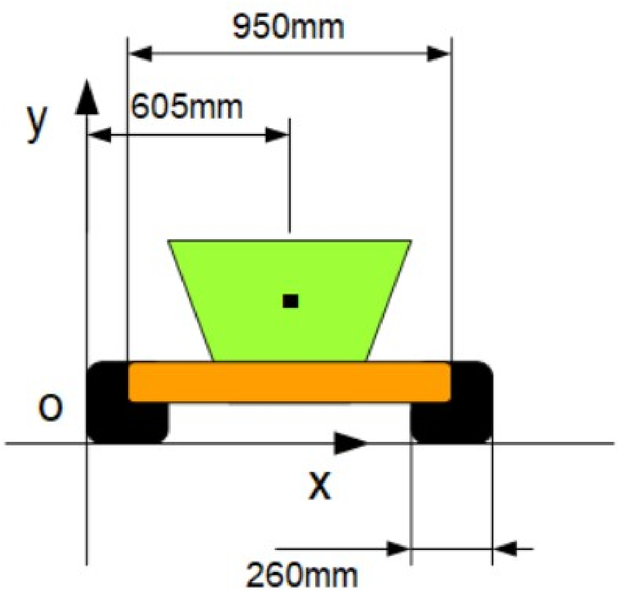


Figure 54: frontal view of the machine

As it can be seen the origin of the coordinates system coincides with the bottom right corner of the right track of the machine. I chose this because it simplifies calculations.

In this reference system the three center of mass of the machine are in the coordinates:

- Motor: x=650,y=820;
- Chassis: x=650,y=200;
- Tool: x=650,y=250.

To obtain the global center of gravity I have to use these formulas:

$$X_{cg}=\frac{\sum_i^n (w_i * x_i)}{\sum_i^n w_i} \quad (22)$$

$$Y_{cg}=\frac{\sum_i^n (w_i * y_i)}{\sum_i^n w_i} \quad (23)$$

then the center of gravity is in position:(650,497).

Then I have to look for the limit angle for turn over.

In an inclined plane the angle of the slope of the plane is the same between the vertical line passing through the center of gravity and the orthogonal to the support plane as represented in the figure below.

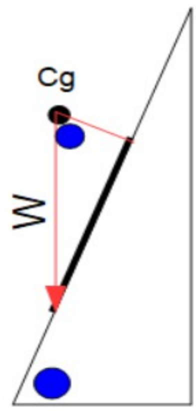


Figure 55: here it can be seen that the two angles marked by the blue dot are the same, the red arrow W represents the weight force and the black line is the support plane

As the slope increases, the vertical line will arrive to the limit of the support base. The limit can be calculated using trigonometry. The vertical passing through the center of gravity, the support plane and the orthogonal to it passing through the center of gravity create a triangle. To calculate the limit I need the measure of the two catethes of this triangle.

6.1.2 Turn over limits

From trigonometric formulas I know that the tangent of an angle is equal to the ratio of the opposite catethus and the adjacent. Then I have to study the ratio between the vertical coordinate of the center of gravity and the length of the support base and confront it with the tangent of the slope. To do so I used Matlab in order to have some help with the calculations, and here it is reported the first result I obtained.

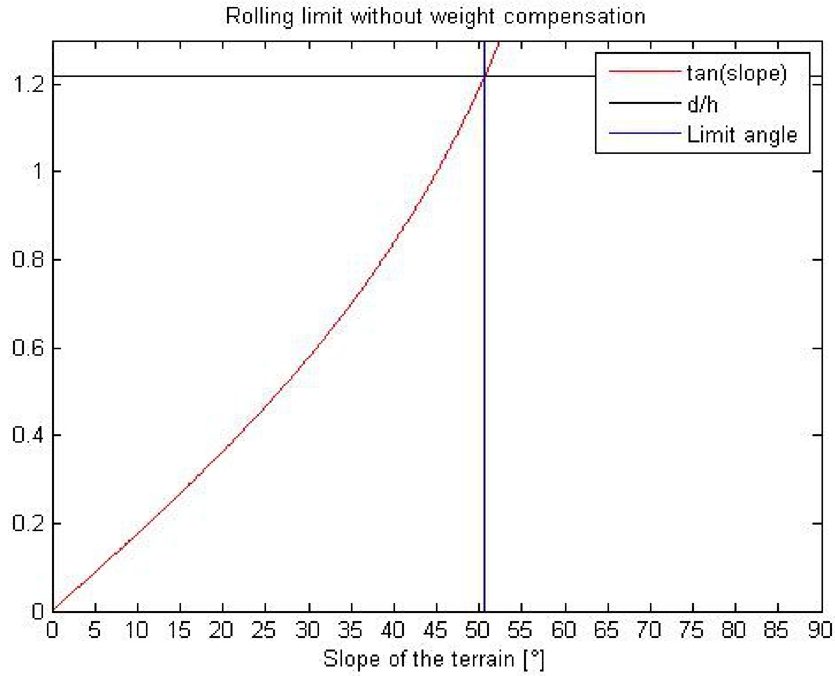


Figure 56: turn over limit for the machine

In the picture it is represented the tangent of the slope of the terrain (red) then is reported the ratio d/h that is the ratio of the x coordinate (length of the support base) of the center of gravity and its y coordinate (the height of the center of gravity). In this case the ratio is:

$$x = 605[mm], \quad y = 497[mm]$$

$$\frac{x}{y} = \frac{605}{497} = 1{,}21 \quad (24)$$

$$atan\left(\frac{x}{y}\right) = 50.42° \quad (25)$$

so if we have a slope of more than 50.42° the machine will turn over. This is the case where the automatic weight compensation system is inactive.

When it is activated things will change. First of all the position of the center of gravity changes. The motor moves and so do its center of gravity. Our machine has two possible movements to do (translate or rotate the motor) and they are done at the same time, so it rotates of a certain quantity then it translates of another quantity. Again I have done some simulation with Matlab and here are the most significant results.

In the picture 57, I represented all the possible positions that can be reached by the center of gravity. The red star is the initial center of gravity, then, as the machine starts to move, the position changes; the height (the y coordinate) reduces while the x increases or decreases with respect to central position.

When the x coordinate reduces it means that we are on a terrain with negative slope. In this case to calculate the right support base we have to change the reference system in a way that the origin now is coincident with the contact point between the right track and the terrain and the x axis is pointing to the left instead of to the right as before. If we do this modification on the reference the calculations are exactly the same.

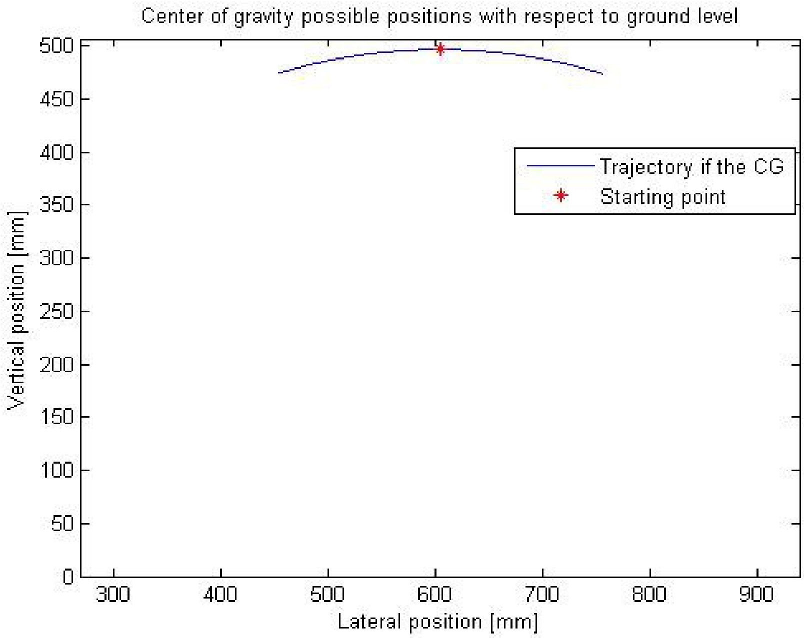


Figure 57: possible position of the center of gravity

As the center of gravity reduces its height and increase its support area the turn over limit is increased.

The maximum length for the support base is reached when the machine is completely translated to the right and totally rotated, this corresponds also to the point of minimum height. In this condition the coordinate of the center of gravity are: 775,78 mm in the x coordinate and 474,16 in the y. By using again formulas 24 and 25 I obtain that the new limit for turn over is 57,89°. In the picture 58 it is represented the new limit (light blue). As expected the balancing of the machine is increased when the compensation is active.

The results of this simulation are shown in the figure 57.

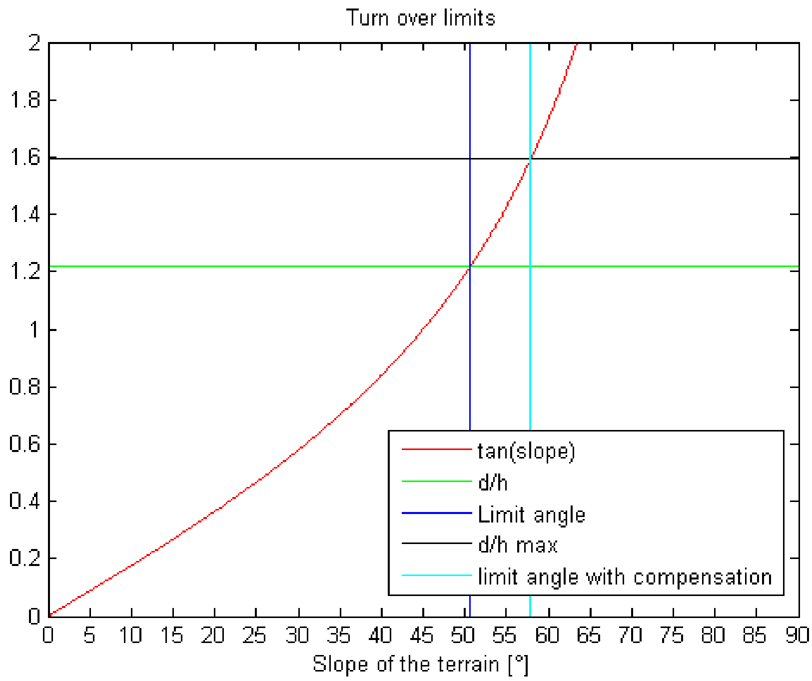


Figure 58: turn over limits

To further increase the stability of this machine, in a future it can be developed with more compact dimensions or it can be equipped with larger tracks.

6.2 Soil compaction

The main goal of the machine is to reduce the soil compaction. This machine is supposed to work not only on open areas but thanks to its compact dimension it can work easily between the rows of a fruit-yard or a wine-yard. These type of cultures, especially the grapes, are usually cultivated on the side of the hills, requiring very skilled tractor drivers and the risk of accident is never zero. Sometimes in the hardest places all the work is done by hand because a tractor can't fit in the rows or it is not secure enough to go there on a machine. Our machine with its characteristics can help a lot the farmers to reduce manual work or risk of accident, since it is supposed to work easily in scenarios like these without the presence of a user next to it.

Another big problem that sorts out during the discussion with some farmers is the soil compaction. The tractors used in a normal scenario are around 2000/2200 kg of weight without any tool attached to it. This machine at most will arrive to weight around 1600 kg, that also includes the weight of the tool on the front and on the rear. The light weight plus the possibility to move it in a way that it is distributed equally on both sides creates a large reduction of the soil compaction.

The movement of the mass of the motor in particular reduces the formation of tracks on the lowest side of the machine.

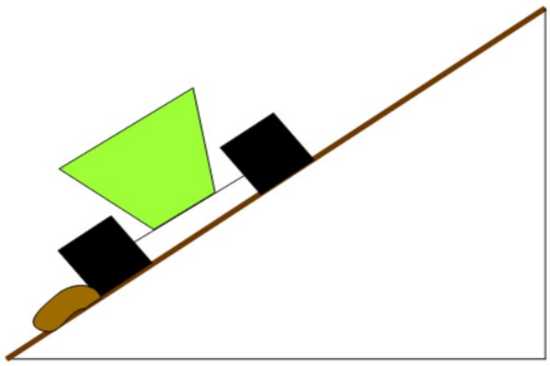


Figure 59: model of the machine on a hill

As it can be seen in the Figure 59 if the balancing system is not active the machine will slip on the side, creating a track (the brown mass). This can create some depression in the terrain that will keep the water and create water stagnation.

But why is it so important to reduce the compaction of the soil? As reported in the article "Investigating the effect of the tractor driving system type on soil compaction using different methods of ANN, ANFIS and step wise regression"(2022, Moinfar) [17] soil compaction creates a lot of problem of chemical and environmental type. On the environment it affects for example the erosion and the capacity of draining water of the terrain. On the chemical side instead the soil becomes harder and so a plant grows its root less than in a normal soil, this will affect the health of the plant and the capacity of production.

To study this phenomenon are required some expansive instruments like a cone penetrometer

that measures the resistance of the soil while a cone is inserted at constant speed in the terrain. Since we didn't have the possibilities to do this experiment, we searched for some theoretical results, one interesting is the one in the picture 60.
In that picture are represented the results of an experiment conducted over different types of terrain and cultures. The experiment consists of measuring with a cone penetrometer the force needed to penetrate the terrain at constant speed. On the x axis we have the force used [N] over cm^2, while on the y there is the depth reached with that force. This study can be a good starting point for this theoretical simulation.
The weight of the machine is around 1600kg, and it is divided over the two tracks whose surface is about $3380cm^2$. The pressure on the soil then is 4,64 N/cm^2
With this force the machine can create a compaction on soil, basing on the theoretical value below, of few cm and only in some types of terrain.
On an inclined plane, the chassis and the tool on the front can be still supposed to be equally distributed on the two tracks since they have very low center of gravity. The motor instead changes a lot the weight distribution.

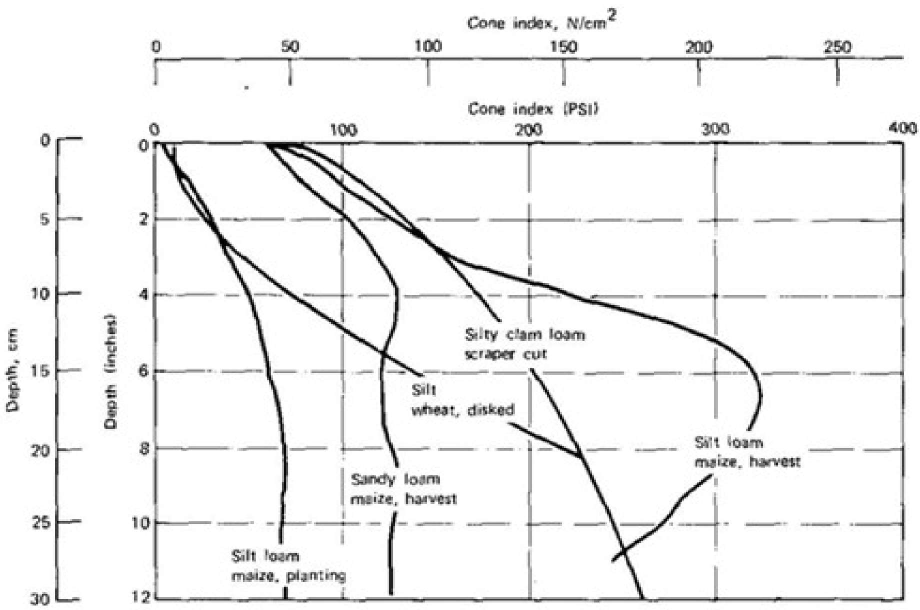


Figure 60: the graph shows the correlation between the pressure on the terrain and the depth of compaction

(from Mechanics of Tillage and Traction 3(2+1),http://ecoursesonline.iasri.res.in/mod/page/view.php?id=2700)[18]

If the balancing system is inactive the weight on the tracks can be very different. At 35° of slope the difference is about 300kg, this is the main reason why the depression cited in picture 38 are formed. If instead the system is active, until we stay inside the 35° area where the system works properly we have almost the same weight on both tracks and then the machine won't slip and the depression isn't formed anymore.

Chapter 7

7.1 Conclusions

As expected, this project wasn't simple. It took lot of trials before arriving to the best solution. We encountered many problems during the first part of the study of the machine. We had to find the possible solutions for having a new machine with something different than the others already present on the market.
In the first trials of the machine we worked in a safe ambient where the cycle of the machine could be stopped quickly if something went wrong. There we saw that the measures were collected in the correct way. Then, once we were sure that the system was working correctly, we moved to a larger area and we tested the whole machine.

Figure 61: first test of the machine

We put a front mulcher made by Meritano, a local producer of farming tools and equipment, on the front, while on the back we mounted a set of weight of around 300kg, and we cleaned an area where the slope on the highest point was higher than 40°.
The system studied worked in a good way. As it started to move on the terrain the position of the motor changed and the machine became more stable.
One problem we encountered during these tests was that the tracks sometimes tended to slip on the grass. We thought then to change the material of the tracks so that it created more grip.

Figure 62: open field test

In conclusion, I can say that the system studied works properly even if we couldn't do any more accurate study of the actual compaction of the soil.
For what regards the possibility of turn over instead, the machine during these tests never showed us a sign of possible turn over even in the steepest areas.
For now we've seen that this machine has lot of potential and also farmers are interested in it. Thanks to this the machine and the system can be upgraded and new technologies can be developed.

7.2 Future works

The machine obviously is not complete, it can have a lot of modifications that only more tests can find out.
One of the goals of the company is to have a simple machine that anybody can use.
One of the biggest issue with these type of machines is the maintenance. Maintenance should be regularly done by the final user but it shouldn't take long time. For this purpose all the most important part of the machine and the motor are in place where they can be easily reached by the operator. Then there are also sensors that are important and may be replaced, in particular the two position sensors. One thing that should be done is to find an autonomous calibration procedure so that if the sensor has to be changed there is no need do it manually.
Another thing that should be done is to increase the possibility to attach instruments to the machine, such that the versatility of the machine is increased. For example providing the machine with a stronger hitch so that some tools for working the soil (like tillers) can be attached.

Another important thing that can be installed on a machine like that is a system for autonomous guide. Some model on the market already have it but are few compared with the total. This can be a very helpful thing for users since if the machine becomes even more automated than now, the amount of work will decrease further more and a single user can control more than one machine at the same time.

Bibliography

1 Energreen site: www.energreen.

2 G.Riccio, "I robot forestali possono piantare migliaia di alberi al giorno" ,2021, https://www.futuroprossimo.it/2021/01/i-robot-forestali-possono-piantare-migliaia-di-alberi-al-giorno/

3 R.A. Awad, M.H. Rais, M. Rogers, I.Ahmed, V.Paquit,"Towards generic memory forensic framework for programmable logic controllers" in Forensic Science International, volume 44, March 2023

4 ifm site. www.ifm.com/it/it/product/CR720S?tab=documents

5 seedstudio site https://www.seeedstudio.com/blog/2019/11/27/introduction-to-can-bus-and-how-to-use-it-with-arduino/

6 E.Aliwa, O.F.Rana, C.Perera, P.Burnap, "Cyberattacks and Countermeasures For In-Vehicle Networks", 2020

7 M.Falch,"canopen-tutorial-simple-intro",2022

8 N.velichkov, "Ultimate CAN-bus guide 2023: a detailed look at the protocol" , on www.autopi.io, January 2023

9."http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/"
10 tlp site: https://www.tsmsensors.com/it/sensori/inclinometri/tlp300-canopen/

11 Codesys site. www.codesys.com

12 S.Sinha, S. Shakya, R.Mukhiya, R.gopal "Design and Simulation of MEMS Differential Capacitive Accelerometer",in VIIth international conference on smart materials, structures and systems, July 2014.

13.Understanding Piezoelectric Accelerometer BasicsMarch 20, 2022 by *Dr. Steve Arar[13]*
14 Omega site. https://www.omega.com/en-us/resources/accelerometers

15 Gary Frigyes, Ed Myers and Jeff Allison , fundametal of fotelctric sensors,(2013), www. Automation.com

16. *https://duplomaticmotionsolutions.com/*

17 A.Moinfar, G. Shahgholi, Y.A.Gilandeh, M.Kaveh, M.Szymonek,“Investigating the effect of the tractor driving system type on soil compaction using different methods of ANN, ANFIS and step wise regression”,in Soil and tillage research, August 2022

18 D.Anantha Krishnan, “Mechanics of Tillage and Traction”, lesson 23, “Cone index and tire basics” http://ecoursesonline.iasri.res.in/mod/page/view.php?id=2700