UNIVERSITÀ DELLA CALABRIA
DIPARTIMENTO DI
INGEGNERIA INFORMATICA,
MODELLISTICA, ELETTRONICA
E SISTEMISTICA
DIMES

Master's Degree course in Automation Engineering

Academic Year 2018/2019

INDUSTRIAL AUTOMATION SYSTEMS

Project Report

Student

Ivonne

Index of Contents

Introduction

The aim of this report is to explain the work that has been done to realize two projects about industrial automation applications.

In the first section, the control and the implementation of a Direct Current motor, equipped on a Laser-Cutting machine has been described. This kind of machinery has been used to make accurate cuts on several materials in industrial environment, such as wood or metal sheet, using laser technology. In this case, a planar laser-cutting machine has been considered, which assigned task has been to cut out the required geometric shapes, respecting a speed parameter that has been established before. So, a control unit to manage two direct current motors has been designed, through MATLAB environment and its Simulink software.

In the next section, a mechanical system used for loading and unloading of goods has been implemented. It has composed by two carts that are able to move themselves back and forth on the rails. This structure has been modelled by Sequential Function Chart, a graphical programming language used for programmable logic controllers, named PLCs.

Chapter 1: Planar Laser-Cutting Machine

1.1. *Description of the machinery*

The machine that has been considered in this work is intended to be used for planar laser-cutting process. In an industrial environment, this kind of operation is highly employed to repeat same tasks over and over, making human supervision unnecessary and obtaining accurate results anyway. Depending on the application and the duration of the assigned process that a planar laser-cutting machine should execute, it needs of a large workspace, which generally isn't smaller than 1 meter by 1 meter.

Figure 1: Planar laser-cutting machine example.

A laser-cutting machine is a Computer Numerical Control (CNC) machinery that adopts a laser source to cut metals, such as steel, aluminium or iron, rubber, wood, plastic and other materials, into 2D or 3D shapes. This is commonly equipped with a machine frame, a laser generator, a laser power supply, a laser cutting tool, a control panel and CNC system.

There are several advantages using this technology, such as laser cutting speed is very fast, that its incision is smooth, has no mechanical stress, high precision, good repeatability and doesn't damage the surface of the material that should be processed.

So, this cutting technique permits to realize very intricate shapes and small holes on different kind of material plate.

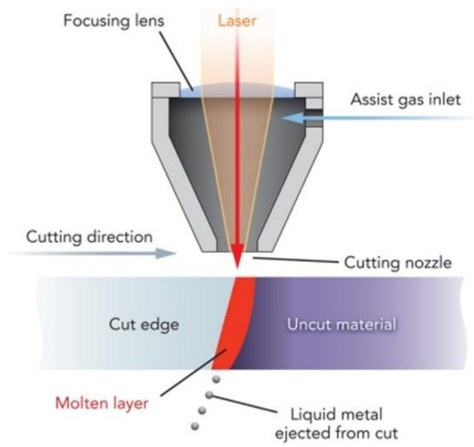


Figure 2: Example of a laser cutting nozzle.

The laser beam is a column of very high intensity light, of a single wavelength. It’s very small in diameter as it travels from the laser resonator, which creates the beam, through the machine’s beam path. The laser direction is regulated by some mirrors, called as “beam benders”, then it goes through the bore of a nozzle, that usually is a compressed gas, and it is finally focused onto the plate. The focused laser beam goes through the bore of a nozzle right before it hits the plate.

1.2. Technical specifications

The kind of machinery that has been considered in this work is a dual axes planar laser-cutting machine, that has equipped on two direct current motors which allow it to operate in a two-dimensional workspace, that it’s been described as a Cartesian space.

The main task of this mechanical system consists in cutting a laminate sheet, to realize the assigned geometric shapes, moving the two axes, one along abscissa direction and the other along ordinate direction, on a rectangular plan, which measures five meters by ten meters. To make this machinery works, some templates must be provided, which represent the requested shapes to cut from the laminate sheet.

These geometric figures are shown below:

Figure 3: Cutting templates.

So, from the laminate, two circular crowns will be extruded, where the internal one has a radius of 0.5 meters dimension and the radius of the external one is equal to 1.5 meters. The other shapes are four semi-circular figures, such as a generic arc, characterised by a height of five meters and a length of 2.5 meters.

To reduce material waste due to the cutting process, the disposal of the geometric shapes has been established before, trying to exploit all the surface of the laminate. Moreover, for this reason, the dimensions of the laminate have been chosen to overlap the whole cutting plan, that is the reachable area by the two axes of the machinery, its workspace.

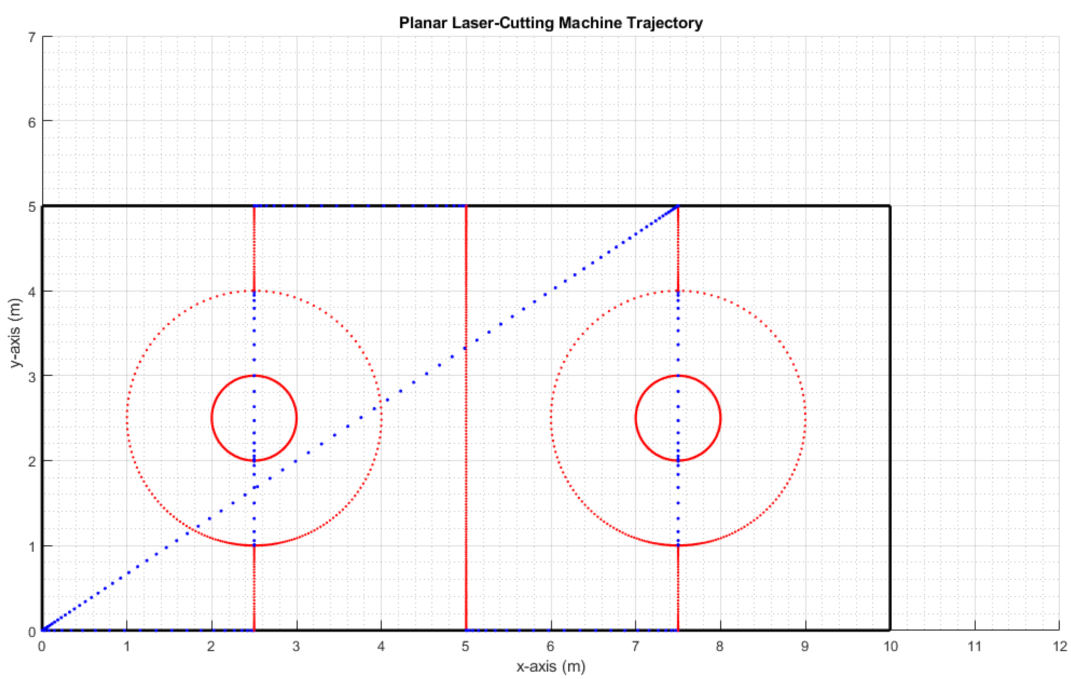


Figure 4: Laminate cutting template.

As shown in the Figure 4, the finished products totally are two circular crowns and four semi-arcs of circumference.

Red lines indicate where the cut has been done, that is when the laser tool has been activated. During the implementation of the control unit, this has been denotated using a Boolean variable as an indicator about the status of laser activity, that it has been set to “1”. When blue lines are figured, this flag has been set to “0” and the laser has been turned off, because the two machine axes should move themselves to reach the next cutting point.

1.3. Trajectory planning

If the position of each geometric shape on the laminate plate has been done to better reduce the material waste, the laser path has been chosen with the aim to decrease the number of movements that the machine should make to accomplish its task. In a manufacturing process, even the time spent to realize finished products it’s important, so the distance between a point of the path and the following one should be minimized as much as possible. About this, a cutting trajectory has been planned to define, point by point, the operations that the machine should perform to cut the designed shapes, moving its axes at assigned mechanical parameters, such as laser tool position, or velocity and acceleration that the two axes should attain during their movements. All these characteristics depend on the technical specifications of the direct current motors that are equipped on this machinery and the way how the path to track it’s generated.

1.3.1. Path generation

The trajectory has been described using polynomial interpolation between two adjacent points, called control points. The kind of approach that has been chosen consists in a spline interpolation, that is a form of interpolation where the interpolant is a piecewise polynomial called spline.

In this case, the advantage is that the interpolation error is smaller than it is in a generic polynomial interpolation, even if a low degree polynomial for the spline has been used.

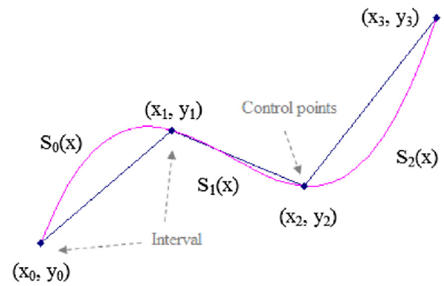


Figure 5: Cubic spline approximation vs exact function.

So, the purpose of this wide class of functions called spline is to interpolate intervals between data points, where the term "interpolation" refers to a spline function that is fitted properly to pass through each point, minimizing some measure of curvature to produce a smooth curve (its first and second derivative are continuous) , when a set of data points has been given before.

Mathematically, a spline function consists of polynomial pieces on subintervals joined together with certain continuity conditions.

Formally, suppose that $n+1$ control points, that are $x_0, x_1, \dots, x_n$ have been specified and satisfy $x_0 < x_1 \dots < x_n$. These points are called knots. Moreover, and integer $k \geq 0$ has been initialized. A spline function of degree k, having knots $x_0, x_1, \dots, x_n$ in a function P such that:

- P is a polynomial of degree $\leq k$, on each interval defined as $[\, x_{i-1}, x_i \,]$;
- P has a continuous derivative of order $(\, k-1 \,)$ on the whole interval $[\, x_0, x_n \,]$;

Therefore, P is a piecewise polynomial of degree at most k if it has all continuous derivatives up to the order $(\, k-1 \,)$.

Hence, a spline of degree n is piecewise polynomial and has the form below:

$$p(t) = a_0 + a_1 t + a_2 t^2 + \dots a_n t^n$$

For this work, the construction of a fifth degree polynomial spline has been preferred to cubic spline interpolation. This choice is due to the purpose to obtain a smoot curve as far as possible, that will approximate the exact function which represents the trajectory to follow. Using this method, a high accuracy and precision in the approximation of the established path has been verified.

To apply a polynomial spline of fifth degree in the trajectory generation, it should be express as a parametric equation of an independent variable that is time, as it has been seen above.

Moreover, to guarantee the execution of a smooth path, a set of constrains on position, velocity and acceleration need to be specified.

This kind of constrains consists in a system of equations which are linear with respect to the coefficients, that are a_0 , ... , a_5 in the case in question. For this reason, the processing of these equations, called boundary conditions, requires low computational capacity. So, to plan a trajectory, that is establishing the way in which the specified movement takes place from the initial position to the final one, the definition of position, speed and acceleration profiles, relating to the assigned boundary conditions, is necessary.

The expression of a generic polynomial spline of fifth degree has been shown below:

$$\lambda(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

Its corresponding boundary conditions are:

$$\lambda(t_i) = \lambda_i \qquad \lambda(t_f) = \lambda_f$$

$$\dot{\lambda}(t_i) = \dot{\lambda}_i \qquad \dot{\lambda}(t_f) = \dot{\lambda}_f$$

$$\ddot{\lambda}(t_i) = \ddot{\lambda}_i \qquad \ddot{\lambda}(t_f) = \ddot{\lambda}_f$$

Where t_i represents the initial time and t_f the final one.

The various profiles that should be calculated applying a series of derivative operations have a form like this:

- $p(t) = a_0 + a_1(t_f - t_i) + \dots + a_n(t_f - t_i)^n$, that is position profile representation;
- $v(t) = \dot{p}(t) = a_1 + 2a_2(t_f - t_i) + \dots + na_n(t_f - t_i)^{n-1}$, that is velocity profile, corresponding to first order derivative of the position;
- $\alpha(t) = \ddot{p}(t) = 2a_2 + 6a_3(t_f - t_i) + \dots + n(n-1)a_n(t_f - t_i)^{n-2}$, which is acceleration profile, i.e. the second order derivative of the position;

After this, the a_i parameters should be calculated.

When the boundary conditions have been established, the a_i parameters should be calculated as shown below, keeping in mind that, in this case, a fifth order polynomial spline is necessary:

$$
\begin{cases}
a_0 = \lambda_i \\
a_1 = \lambda_i \\
a_2 = \frac{1}{2}\ddot{\lambda}_\imath \\
a_3 = \frac{1}{2\left(t_f - t_i\right)^3}\left[20\left(\lambda_f - \lambda_i\right) - \left(8\dot{p}_f + 12\dot{\lambda}_\imath\right)\left(t_f - t_i\right) - (3\ddot{\lambda}_f - \ddot{\lambda}_\imath)\left(t_f - t_i\right)^2\right] \\
a_4 = \frac{1}{2\left(t_f - t_i\right)^4}\left[30\left(\lambda_f - \lambda_i\right) + \left(14\dot{p}_f + 16\dot{\lambda}_\imath\right)\left(t_f - t_i\right) - (3\ddot{\lambda}_f - 2\ddot{\lambda}_\imath)\left(t_f - t_i\right)^2\right] \\
a_5 = \frac{1}{2\left(t_f - t_i\right)^5}\left[12\left(\lambda_f - \lambda_i\right) - 6\left(\dot{\lambda}_f + \dot{\lambda}_\imath\right)\left(t_f - t_i\right) - (\ddot{\lambda}_f - \ddot{\lambda}_\imath)\left(t_f - t_i\right)^2\right]
\end{cases}
$$

Where the constraints at points $\lambda(t_i)$ and $\lambda\left(t_f\right)$ can be defined in a matrix form as follows:

$$
\begin{bmatrix}
1 & t_i & t_i^2 & t_i^3 & t_i^4 & t_i^5 \\
0 & 1 & 2t_i & 3t_i^2 & 4t_i^2 & 5t_i^4 \\
0 & 0 & 2 & 6t_i & 12t_i^2 & 20t_i^3 \\
1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\
0 & 1 & 2t_f & 3t_f^2 & 4t_f^2 & 5t_f^4 \\
0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3
\end{bmatrix}
\begin{bmatrix}
a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5
\end{bmatrix}
=
\begin{bmatrix}
\lambda_i \\ \dot{\lambda}_\imath \\ \ddot{\lambda}_\imath \\ \lambda_f \\ \dot{\lambda}_f \\ \ddot{\lambda}_f
\end{bmatrix}
$$

The relative boundary conditions are:

$$\lambda(t_i) = a_0 + a_1 t_i + a_2 {t_i}^2 + a_3 {t_i}^3 + a_4 {t_i}^4 + a_5 {t_i}^5 = p(t_i)$$

$$\lambda(t_i) = a_1 + 2a_2 t_i + 3a_3 {t_i}^2 + 4a_4 {t_i}^3 + 5a_5 {t_i}^5 = v(t_i)$$

$$\lambda(t_i) = 2a_2 + 6a_3 {t_i}^3 + 12a_4 {t_i}^4 + 20a_5 {t_i}^5 = \alpha(t_i)$$

$$\lambda\left(t_f\right) = a_0 + a_1 t_f + a_2 {t_f}^2 + a_3 {t_{if}}^3 + a_4 {t_f}^4 + a_5 {t_f}^5 = p(t_f)$$

$$\lambda\left(t_f\right) = a_1 + 2a_2 t_f + 3a_3 {t_f}^2 + 4a_4 {t_f}^3 + 5a_5 {t_f}^5 = v\left(t_f\right)$$

$$\lambda\left(t_f\right) = 2a_2 + 6a_3 {t_f}^3 + 12a_4 {t_f}^4 + 20a_5 {t_f}^5 = \alpha\left(t_f\right)$$

The following graphs represent the behaviour of a generic polynomial spline of fifth degree, in terms of position, velocity and acceleration profiles:

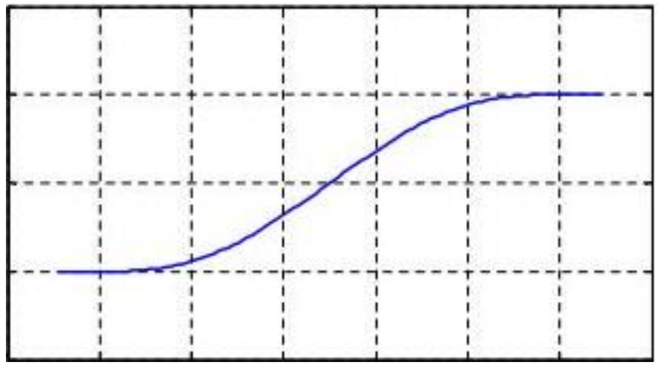


Figure 6: Example of fifth order spline position profile.

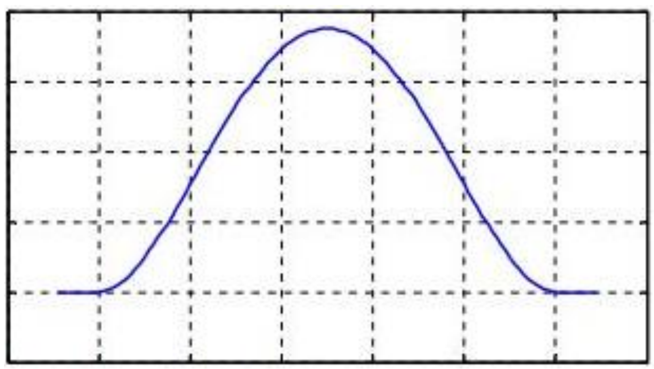


Figure 7: Example of fifth order spline velocity profile.

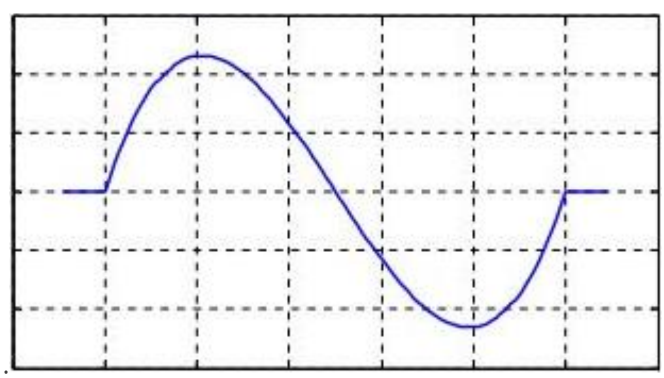


Figure 8: Example of fifth order spline acceleration profile.

At this point, to use the approximation of the fifth order polynomial spline, the time should be discretized.

So, time variable has been parameterized as $0 \leq \sigma \leq 1$, where “sigma” has been defined such as:

$$\sigma = \frac{t - t_i}{t_f - t_i}$$

This parametrization permits to describe the behaviour of a physical system in terms of its motion as function of time, indeed equations of motion can be obtained to act trajectory planning.

1.3.2. Reference generation

The polynomial spline that has been chosen to path generation is a fifth order function defined as `lambda=@(o)(o.^3.*(6.*o.^2-15.*o+10))`.

To cut all the requested pieces of the provided laminate, each template of the geometric shape that should be realized, has been described by a parametric expression, that has been calculated through the corresponding MATLAB script.
Both the function that generates a segment and the function that creates a circumference work receiving as input parameters the initial and the final instant of time, of which the first refers to the beginning of the movement and the second is relative to the time at the movement stops.

The segment parametric function generator has been transcribed below:

```
function [ xf_m, yf_m] = segment(xi, yi, ti, tf, xf, yf)
% [x final movement, y final movement]
global lambda speed cutting_speed dt
% segment between (xi;yi) and (xf;yf) points has been create
t = ti:dt:tf;
% time parameterization
o =((t-ti)/(tf-ti));
% segment generation
xf_m = xi + lambda(o)*(xf - xi);% movement along x axis
yf_m = yi + lambda(o)*(yf - yi);% movement along y axis
end
```

where the others input parameters are the initial and the final position, in terms of coordinates, on the laminate surface.

Moreover, this is the circumference parametric function generator:

```
function [ xf_m, yf_m] = circumference(ti, tf , Ox, Oy, r)
% [x final movement, y final movement]
% Ox and Oy are the coordinates of the center of the circle
global lambda dt
t = ti:dt:tf;
% time parameterization
o =((t-ti)/(tf-ti));
% circumference generation
fCx=@(t,R)(Ox+R*cos(t));
fCy=@(t,R)(Oy+R*sin(t));
%angle range from 3/2 pi until 3/2 pi (360°)
theta = 3/2*pi + lambda(o)*(2*pi);
% trajectory generation
xf_m = fCx(theta,r);
yf_m = fCy(theta,r);
end
```

which receives as input parameters the origin of the circumference that should be cut from the material, and its radius dimension.

It has been assumed that the sampling step used for the discretization of the points interpolating the desired trajectory is equal to $dt = 0.1$.
This value has been demonstrated to be a good compromise to obtain a high accuracy in the planned trajectory tracking and in terms of time computation, too.
It has been noticed that the number of samples, i.e. the control points, increases drastically during the cutting phase whereas the required time for the interpolation operation considerably increases.

Example of code that has been used to implement the cutting phase, that is when laser beam is on:

```
%% Point 3
% from the current point, the first circle has been cut,
% that is the external one.
%circumference C1
%(2.5,1) is the starting point and it's also the ending point,
% when the whole circumference has been outlined.
x3 = x2;
y3 = y2;
% Ox12 and Oy12 are the center coordinates
% circumference motion equation
C1 = 2*pi/r13;
% next time step
t3 = t2 + (C1)/cutting_speed;
%trajectory
[x3_m, y3_m] = circumference( t2, t3 , Ox12, Oy12, r13);
%ending point in terms of x and y coordinates
x_movement = [x_movement, x3_m];
y_movement = [y_movement, y3_m];
%laser = 1, the cut has been made
laser = [laser,ones(1,length(x3_m))];
```

Now, the laser flag value has been set to “1”, because of its status is “ON”.

Example of code that has been used to implement the moving phase, that is when laser beam is off:

```
%% Point 4
% After that circumference has been cut, the machinery
% moves from (2.5,1) to (2.5,2)
% without executing the cutting operation
%vertical line
x4 = x3; % unchanged
y4 = 2;
% next time step
t4 = t3 + (y4-y3)/speed;
%trajectory
[x4_m, y4_m] = segment(x3, y3, t3, t4, x4, y4);
%ending point in terms of x and y coordinates
x_movement = [x_movement, x4_m];
y_movement = [y_movement, y4_m];
%laser = 1, the cut hasn’t been made
laser = [laser,zeros(1,length(x4_m))];
```

In this case, the laser flag value has been set to “0”, because of its status is “OFF”.

According to technical specifications, during cutting phase laser beam has been activated, so the velocity at the machinery joints move is equal to *cutting_speed* parameter that is 0.2 m/s. This value is due to the *reduction factor* α that characterized each direct current motor that has equipped on each of the two prismatic joints that permit the machinery movement through the whole workspace.

When laser beam has been turned off, any part of the laminate shouldn’t be cut off, so one of the motors or both move themselves to place the laser tool on the next control point starting from which the machine should resume the cutting phase. During this operation, the motor speed isn’t subjected to any reduction, so the value of *speed* parameter is 1 m/s.

All the movements that the planar laser-cutting machine acts have been shown through a plot, by a MATLAB script named “*trajectory_plot”*.

When the reference generator has finished to compute position, velocity and acceleration profiles which characterize the planned trajectory, these values have been passed to Simulink software, by the MATLAB script named *“reference_values”* which makes use of a MATLAB proprietary function that is “*timeseries*”.

The data set in question consists in succession of points which define the position that the two machinery axes should assume, during the passage of time, to outline the specified geometric shapes.
Starting from the interpolated points, that have been expressed in terms of Cartesian coordinates in the x-y space, it’s also possible to obtain the constrains on speed and acceleration, in addition to the laser status.

It has been specified that all the phases of the development of this project will be executed running the code implementation in the MATLAB script called as “*main*”.

1.4. Control structure of a direct current motor

In this section of the work, the control of the two direct current motors that are equipped on the planar laser-cutting machine has been implemented, using Simulink software. Then, a simulation process has been executed to demonstrate that the regulator that has been designed is able to make sure the planar laser-cutting machine completes its task. The term “regulator” refers to a device which has the function of maintaining a designated characteristic. For example, it can be utilized to perform the activity of managing or maintaining a range of values in a machine during its work. So, it can be said that it’s able to solve a control problem that, in the case in question, consists in an operation of trajectory tracking, where in addition to position regulation, also the constrains on velocity and acceleration should be satisfied.

1.4.1. DC Motor

DC motors belong to the class of electric actuators that are devices for the conversion of energy from electric domain to the mechanical one. Generally, they implement a transduction of power and signals for motion generation. The most common types rely on the force that is produced by magnetic fields. DC motors were the first widely used form of motor, as they could be powered by existing DC power distribution systems. However, the advent of power electronics has made it possible to replace them with AC motors in many applications.

The first DC motor has been developed around the 1830’s-1840’s but was commercially unsuccessful, because this kind of motors have been powered by poor batteries, even if they have been expensive. Later, the first viable DC motors have entered the market in the late 1800’s due the invention of the electrical grid, that is an interconnected network for delivering electricity from the electrical power generating stations to distant consumers, and the possibilities to build rechargeable batteries. Consequently, brushed DC motors have been continuously improved and even if they have been considered of low utility today, because of the development of other types of electromotors, such as brushless DC motors and induction motors, they are still used in many applications, and in effect they appear in a large variety of shapes and sizes. A fundamental characterization of DC motor is brush type or brushless ones. This refers to the manner of commutation used in motor, which converts direct current from the power supply into the alternating current that is required to generate motor action. Therefore, if this commutation is performed mechanically with brushes, the commutator segments at the ends of the rotating rotor coil physically slide against the stationary brushes that are connected to the terminals of motor and so this type of motors is called Brushed DC motor. For brushless motors, the DC is converted into AC in the rotor electronically, with position sensors and a microprocessor controller, so no brushes are needed.

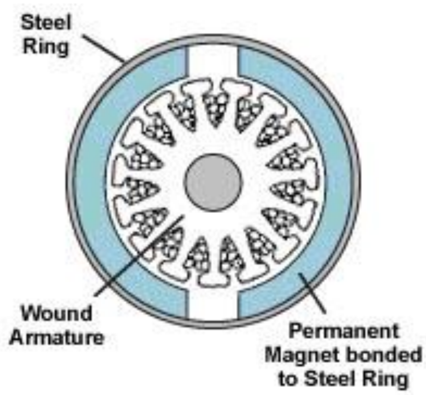


Figure 9: Brushed DC motor.

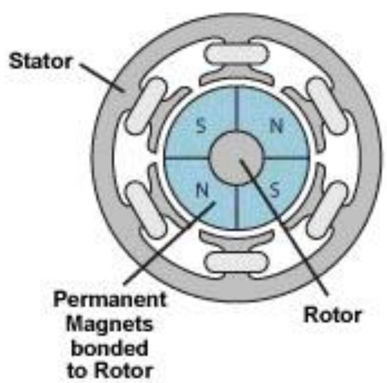


Figure 10: Brushless DC motor.

The two motors that are equipped on the planar laser-cutting machine that is considered in this work, are two DC brushed motors.

A conventional DC motor is formed by an arrangement of coils that creates motion from electrical power. It consists of two main parts, the stator and the rotor. The first is composed by of either a permanent magnet or electromagnetic windings and creates a stationary magnetic field around the rotor, which occupies the central part of the motor. It's made up of one or more electric windings around armature arms, that generate a stationary magnetic field when energized by the external current. Then, the magnetic poles thus generated by this rotor field are attracted to the opposite poles, that are generated by the stator field and repelled by the similar poles. This situation causes the rotor to rotate.

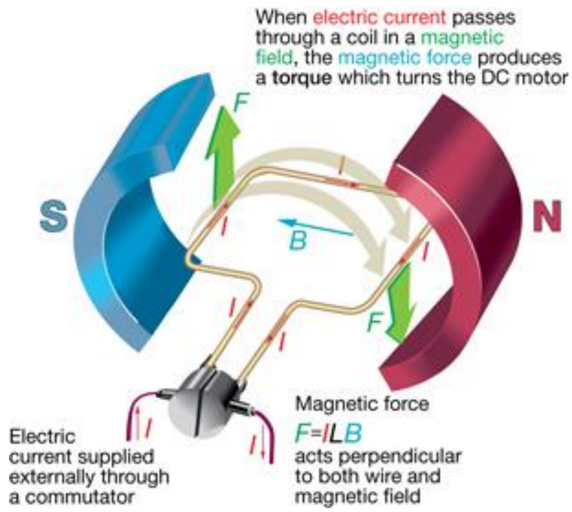


Figure 11: DC motor working.

The rotation continues due to the commutator. Basically, this commutator manages the direction of the flow of current and thereby the direction of the magnetic field, as shown in the image above. When the rotor turns due to attraction and repulsion actions and the rotor becomes horizontally aligned, both brushes contact the opposite side of the commutator. In this way, the current through the rotor is reversed and as consequence, also the magnetic field is reversed. This process repeats itself until the power is supplied to the DC motor and the mechanical action of switching of the filed in the rotor windings is called "Commutation".

Generally, the speed of a DC motor can be controlled over a wide range, using a variable supply voltage or changing the current force in its field windings.

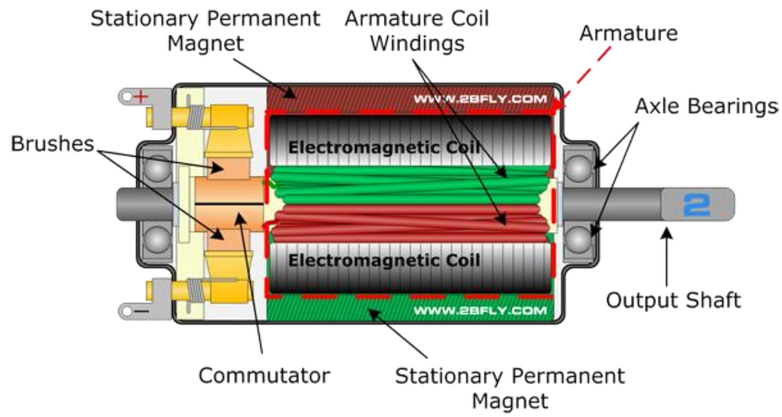


Figure 12: Brushed DC motor model.

Moreover, if the shaft of a DC motor is turned by an external force, the motor will act like a generator and produce an Electromotive force (EMF). During normal operation, the spinning of the motor produces a voltage, known as the counter-EMF (CEMF) or back EMF, because it opposes the applied voltage on the motor.

So, for what concerns DC motor physics, the force that causes the rotor movement is called "Lorentz Force", that is the force that is exerted on an electrically charged object due to an electromagnetic field. This happens because if an electron is moving through a magnetic field, it experiences a force. For this reason, if a current passing through a wire in a magnetic field, the wire experiences a force that is proportional to the cross product of the current itself, expressed as a vector, the magnetic field in question, and the length of the conductor, as follows:

$$\vec{F} = \vec{B}\; x\; l\; x\; \vec{\imath}$$

Now, a straight conductor has been considered, that is traversed by a current i arranged perpendicularly to the lines of force of a uniform magnetic field B. If the conductor has been constrained to rotate only around a specific axis that is perpendicular to the lines of the force of the magnetic field, maintaining the current direction always perpendicular to it, magnetoelectric forces have been generated in this way. They have been characterized by maximum modulus values, so they can give rise to a couple that tends to rotate the conductor in question. Obviously, its direction of rotation depends on the direction of the current i and of the supply voltage V_a.

Because of the magnetic field, the commutating plane changes its position. Then, when it's been affected by magnetic field effect, the commutating plane has been characterized by assuming an idealized position, that isn't subjected to any distortion, as shown in Figure 13.

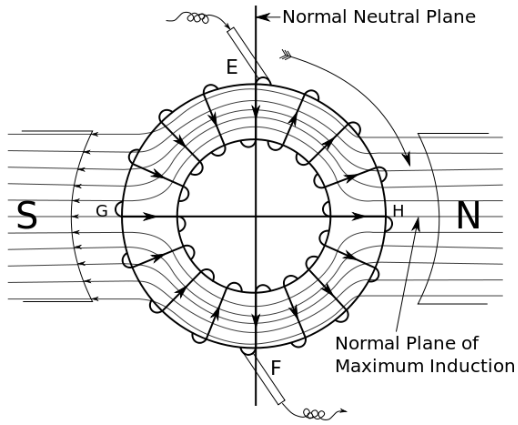


Figure 13: Position of commutating plane subjected to any field distortion effects.

In the case that the compensation for the stator field distortion has been occurred, the position of the commutating plane has been characterized by the necessity to compensate these distortions, so the ideal position has been abandoned, as shown in the following figure:

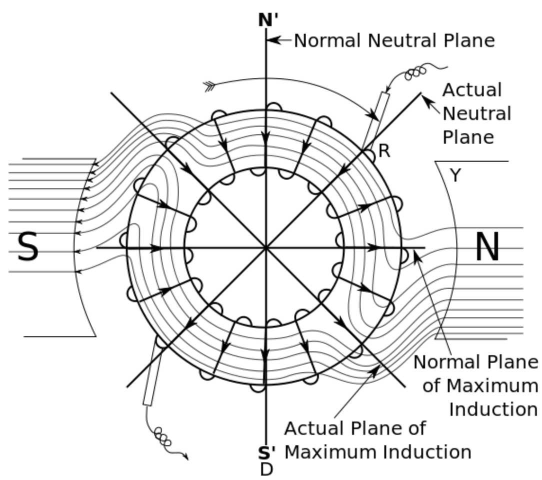


Figure 14: Position of commutating plane subjected to field distortion effects.

Generally, in a Direct Current motor, the stator is made of ferromagnetic material, that is the magnet permanent. This is the inductor of the machine and generates the main magnetic field characterized by the magnetic flux Φ_e , that is named as "excitation flux". Furthermore, the rotor, that consists in a cylinder placed inside the stator, is the armature of the motor. The rotor winding, that is called "armature circuit", consists of a series of bobbins or coils placed inside appropriate channels formed along the generators (or the directrices) of the cylinder. This circuit is powered by an armature current, transferred thanks to fixed contacts, that are brushes, crawling on slats integrated with the rotor, that is called “collector”. In this way it’s possible to maintain the direction of the armature current i_a always perpendicular to the direction of the excitation flow Φ_e . In general, the excitation flow can also be generated by a second electromagnetic circuit called “excitation circuit”, where the stator has poles, named “inductor poles”, on which some coils are wrapped and powered by the excitation voltage V_e . So, to control the motor, it’s possible to act both on the armature voltage V_a, changing i_a value, and on the excitation voltage V_e, modifying Φ_e.

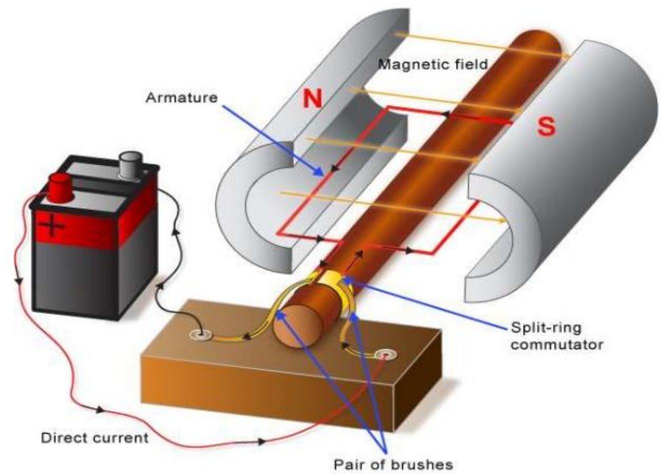


Figure 15: Direct current motor scheme.

In both versions, if the motor is made of a permanent magnet stator, the **electric equation** of the armature circuit is of the following type:

$$V_a(t) = Ri_a(t) + L\frac{di_a(t)}{dt} + e(t)$$

Where $V_a(t)$ is the *Voltage* through the coil, L is an *Inductance* in series with a *Resistance* R, that are the parameters which describe the armature, and both are in series with $e(t)$, an induced voltage that opposes the voltage source. This *back electro-motive force* is due to the rotation of the electrical coil through the fixed flux lines of permanent magnets, by *Lenz law*.

The differential equation relating the armature current i_a and the back e.m.f. e to the armature voltage V_a can be obtained by applying *Kirchhoff’s Voltage Law*. It says that, at any given instant of time, the algebraic sum of voltages around any loop in any electric network is zero.

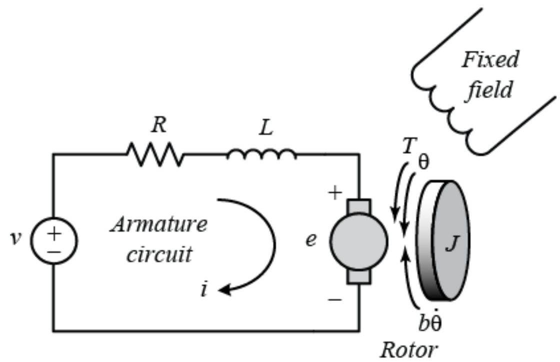


Figure 16: Direct current motor system.

The back emf voltage $e(t)$ is proportional to the angular velocity ω(t) of the rotor in the motor and it's expressed as:

$$e(t) = k_e\dot{\theta}(t) = k_e\omega(t)$$

Where k_e the back electromotive force constant and ω the armature angular velocity of the motor shaft.

In addition, the motor generates a torque τ proportional to the armature current, that is the **electro-mechanical coupling relation**:

$$\tau(t) = k_m i_a(t)$$

Where k_m is torque constant, that depends on given motor.

The term "*torque*" refers to the torque moment along an axis and is measured from the product of force and radius in a right angle with the action of the force. It has been defined as Torsion by the following relation:

$$\tau = F\ x\ r$$

Where r is the position vector, that has been defined from the origin of the coordinate system to the application point of the force vector F. In the case of an electric motor, the output torque depends on the intensity of the magnetic fields of the rotor and the stator and in their phase relationship:

$$\tau = B_r\ x\ B_s\ x\ \sin\Theta$$

Where B_r is the rotor magnetic field, B_s the stator one and Θ represents the angle between two magnetic fields, that is named "load angle". The maximum value of torsion, and therefore the maximum efficiency, is obtained with a load angle of 90°. So, ensuring this value, the relation can be written as it has been done before, knowing that:

$$B_r\ x\ B_s\ x\ \sin\Theta = B_r\ x\ B_s = B = k_m i_a(t)$$

that is the torque τ expression.

If the magnetic field is constant, the motor torque increases linearly with the armature current. So, τ is proportional to only i_a by a constant factor k_m .

Then, two expressions concerning power will be defined below:

- $P_e(t) = e(t)i_a(t)$, that is the electrical power supply;
- $P_m(t) = \tau(t)\dot{\theta}(t)$, that is the mechanical generated power;

Assuming the principle of power has been verified, all the power is transferred from the source to the destination, such as from the input of the system to its output, without any kind of power dispersion. At this point, the two expressions can be matched as follows:

$$P_e(t) = P_m(t) \rightarrow e(t)i_a(t) = \tau(t)\dot{\theta}(t) \rightarrow$$

$$k_e i_a(t)\dot{\theta}(t) = k_m i_a(t)\dot{\theta}(t) \rightarrow$$

$$k_e = k_m$$

In effect, in the International System of Units, the motor torque and back emf constants are considered as equal and so are both indicated as k.

So, k_e is basically considered the inverse of k_m , as $k_e k_m = 1\left[\frac{rad}{s*V}\frac{Nm}{A}\right]$. This happens because motor constants result, generally, from the construction detail of a given motor, such as active length and diameter, winding geometry, resistance and number, magnetic flux densities and permanent magnets.

At this point, in addition to the electrical part, also the mechanical one, which is related to the rotor, should be considered. Referring to the Newton's second law of motion, the rotational motion can be described by the **mechanical equation** below:

$$\tau(\mathrm{t}) = k i_a(t) \rightarrow$$

$$k i_a(t) = J\ddot{\theta}(t) + b\dot{\theta}(t)$$

Where J is total load inertia reflected at the motor shaft and b is rotational viscous friction constant.

Finally, the equations that represent a DC motor behaviour are:

$$\begin{cases} J\ddot{\theta}(t) + b\dot{\theta}(t) = k i_a(t) \\ R i_a(t) + L\dfrac{d i_a(t)}{dt} = V_a(t) - k\dot{\theta}(t) \end{cases}$$

Taking Laplace transformation to them and resolving this system in the frequency domain, the Direct Current motor transfer function has been obtained.

After some algebraic manipulations of the following expressions:

$$\begin{cases} k i_a(s) = J s^2 \theta(s) + b s \theta(s) \\ V_a(s) = R i_a(s) + L s i_a(s) + k s(s)\theta(s) \end{cases}$$

The **motor transfer function** is expressed as an input-output form, that is:

$$G(s) = \frac{\theta(s)}{V_a(s)} = \frac{k}{JLs^2 + (RJ + bL)s + (bR + k^2)}$$

The direct current motor scheme is the following:

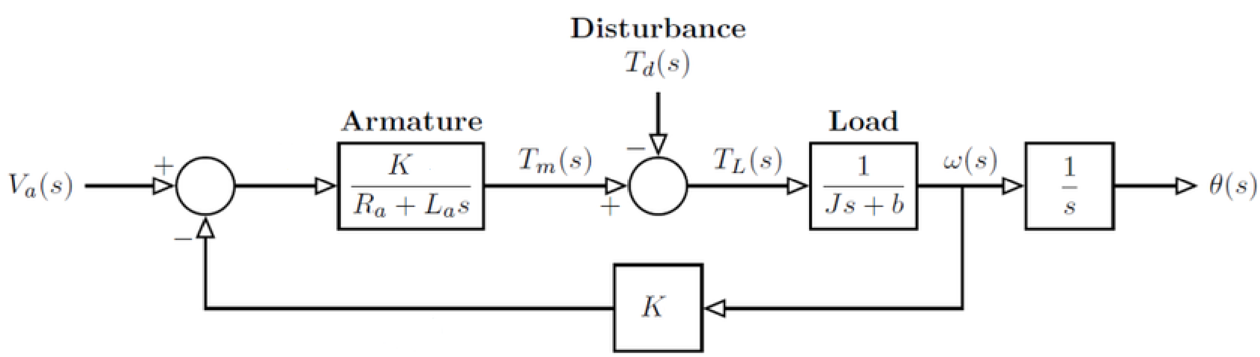


Figure 17: Block diagram of a permanent DC motor.

Any electric motor may be described in terms of two blocks, that represent:

- Electro-magnetic part, taking as input the voltage and giving as output the torque;
- Mechanical part, taking as input the torque and giving as output the velocity;

So, the electrical subsystem of the motor is therefore characterized by first order dynamics, having a real pole equal to $p_e = -\frac{R_a}{L_a}$, named **electric pole** and the mechanical subsystem that is of the first order too, characterized by the pole $p_m = -\frac{b}{J}$, that is called **mechanical pole**.

In general, the electric pole is much faster than the mechanical one, in fact it has a constant of time τ_e in the order of millisecond, instead the time constant τ_m of the mechanical pole is usually a few seconds.

Using Simulink software, a block diagram of a direct current motor has been implemented:

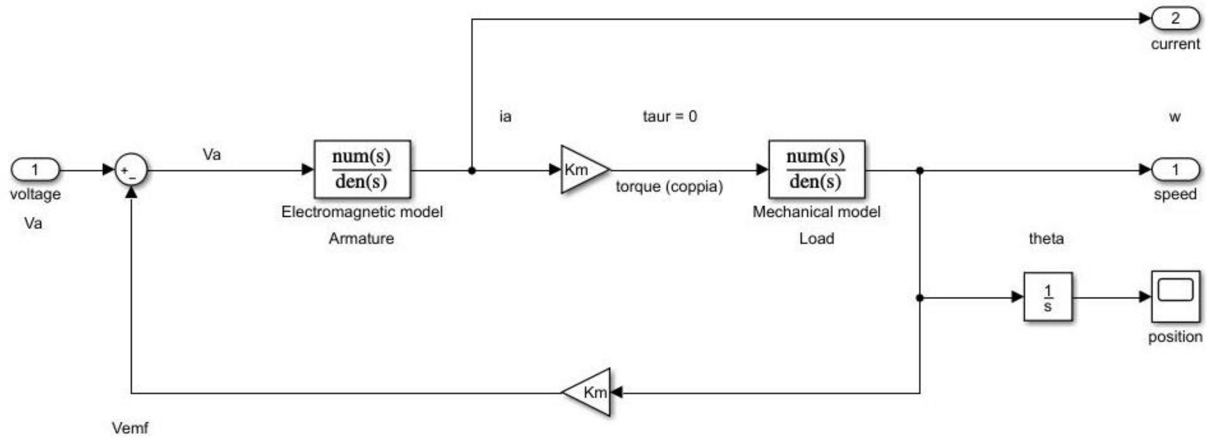


Figure 18: Simulink motor open-loop block diagram.

1.4.2. Control System characteristics

Controlling a DC motor means to apply an appropriate voltage $V_a(t)$ so that the motor tracks, satisfying certain specifications, a desired position reference, rejects the unknown torque τ_r , that in this work has been considered null and maintains its stability.

The strategy adopted for controlling the motor consists in compensating the counter-electromotive force and implementing a cascade structure, composed of:

- Current control loop, that is the internal loop, represents the control of electrical dynamics with feedback of the armature current;
- Velocity control loop, that is the central loop, consists in control of mechanical dynamics with feedback of the angular speed of the motor;

- Position control loop, or the external loop, that is control of the integral dynamics with feedback of the angular position of the motor;

In this way it's possible to divide the assigned control problem into three subproblems that are simpler than the first. In fact, the only constrain is that the frequency separation has been required, for the three control loops, to avoid any interference, so the three controllers should be separately designed.

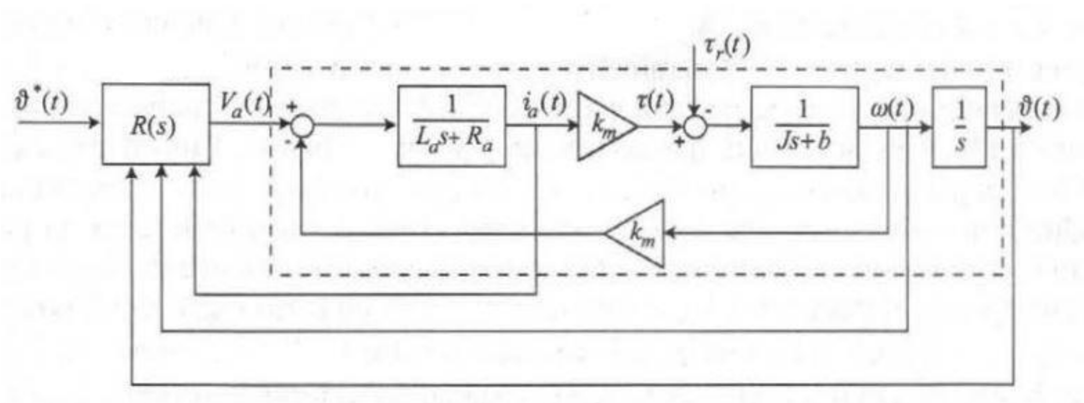


Figure 19: DC motor control scheme.

At these standard feedback loops, some feedforward actions have been often added.

"Feedforward" is a term describing an element, within a control system, that is often a command signal from an external source. A control system that has only feed-forward behaviour reacts to its control signal in a pre-defined way, without responding to how the load changes. In this way, the output isn't adjusted taking account of the load, that is considered, on the contrary of a feedback control system, to belong to the external environment. This means that the control action isn't based on error but only on the mathematical model which describes the process in question and on measurements of any disturbances. In conclusion, a feedforward action involves predictive processes related to the way of obtaining the desired results.

In this work, a feedforward action consists, for example, in the compensation of counter electromotive force, using the measurements of the angular velocity to remove the intrinsic feedback of the motor. So, the term $e(t) = k_e\omega(t)$ is added to compensate the back emf effect, that is often a dominant value in the armature voltage, but to do this action the knowledge of the motor velocity $\omega(t)$ is necessary, and even an estimation of it is useful. All the feedforward actions, as the one that has been explained above, or the feedforward current or voltage, have been computed using the reference values of position, velocity and acceleration of the system dynamics in question. In this way, when a desired trajectory has been calculated, also the corresponding velocity and acceleration are available. So, the feedforward current $i_{ff}(t)$ can be computed using desired acceleration.

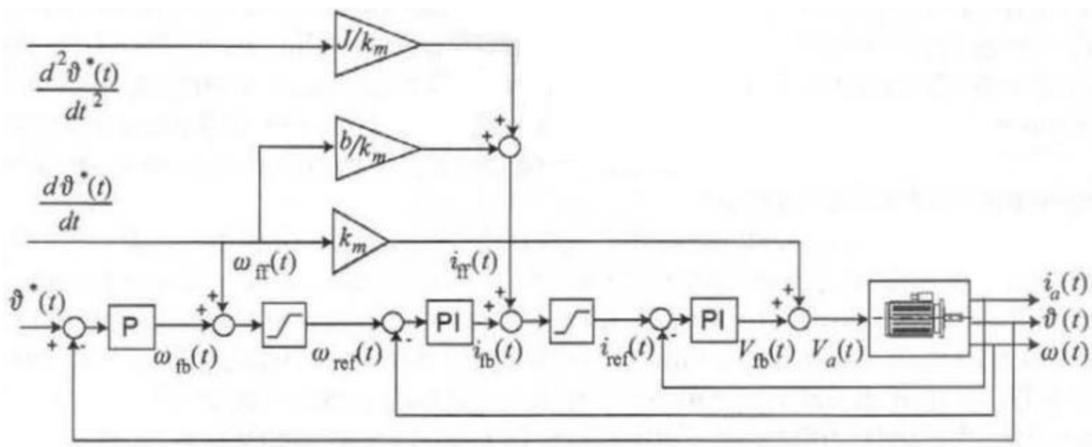


Figure 20: DC motor controller scheme.

Regarding the project design, the *dynamic separation principle* permits to considered negligible the dynamics of the inner loop. This means the inner loops should be "faster" than the more external loops, so the current loop bandwidth must be higher than the velocity one and the bandwidth of the velocity loop must be higher than the position loop bandwidth. Moreover, the bandwidth of the current loop is limited by bounds on the input voltage, due to technological limitations related to measurement noise and sampling frequency. Once that the separation of the three dynamics has been guaranteed, the position loop should be designed, considering the inner loops as ideal electric actuators. Then, the position controller will generate a speed reference, called ω_{ref} , that will be the input of the integrator, which is the position controller. It has been designed as a proportional regulator, to ensure asymptotic stability and null error at steady state. Furthermore, it's advisable to insert also a saturation block that limits the speed command to the physical limits imposed by the motor.

The position loop transfer function is:

$$G_p = \frac{1}{s}$$

For what concerns the velocity loop controller, it's characterized by the mechanical dynamics of the motor system and it's realized as a proportional and integral regulator, using feedback action, to guarantee null error at steady state for constant references and disturbances.

The velocity loop transfer function is:

$$G_\omega = \frac{k}{Js + b}$$

Then, the saturation of the current command needs to be inserted, to maintain its value into the range described by the physical limits of the system. Whereas the feedback controller contains an integral action, as in the case of position loop, the saturation should be managed a using anti-windup scheme. The application of an anti-windup strategy permits to prevent the controller from going into deep saturation and to check windup, that is an anomalous non-linear overshoot of controller output, for example using a desaturation of the integral action to avoid the system instability.

The inner loop is the current one and it can be designed considering only the electric dynamics of the motor system. The control action consists of the compensation of the counter electromotive force and of a regulator in feedback on the current value that is generated by the velocity loop. Generally, this is a proportional and integral controller, designed to maintain the desired pass-through bandwidth and margin of phase, guaranteeing also robustness, even if managing the physical saturation of the voltage command is necessary by another anti-windup scheme.

The current loop transfer function is:

$$G_i = \frac{1}{L_a s + R_a}$$

At the end, the open loop control actions should be designed, by the plan inversion, to ensure an optimal tracking of the position and velocity values, supplied by the reference generator.

These feedforward actions of position and velocity respectively are:

$\omega_{ff}(t)= \left(\frac{1}{s}\right)^{-1} \hat{\theta}(t)$, where $\hat{\theta}$ is the target position and $i_{ff}(t)= \left(\frac{1}{Js+b}\right)^{-1} \widehat{\omega}(t)$, in which $\widehat{\omega}$ is the target value of angular velocity of the rotor.

For these control actions to be realizable, the trajectory of reference $\hat{\theta}(t)$ must be continuous, limited, with first derivative that is continuous and limited.

As it has been said before, the actuation devices, as the PID controllers, have physical limitations due to which they are unable to provide a control signal that is higher or lower than the values belonging to a specified range. So, it’s necessary to use a saturation function, that is described in this way:

$$saturation(u(t)) = \begin{cases} u_l, & u(t) \geq u_l \\ u(t), & u_l \leq u(t) \leq u_h \\ u_h, & u(t) \leq u_h \end{cases}$$

Regarding the interval of admissible values, u_l is the lower bound and u_h is the upper bound. When the control signal belongs to this range, then it has been transferred correctly from the controller to the plant. On the contrary, if the signal control exceeds, the regulator provides a control action equal to one of its limits, without taking account of the system input and so causes an open loop behaviour and the wind-up phenomenon.

1.4.3. Control system implementation and simulation

To test that Direct Current motor controller works properly, a simulation has been executed. Motor parameters used to realize it are reported below:

- $R_a = 1{,}025\ \Omega$, armature resistance;
- $L_a = 0.0001\ H$, armature inductance;
- $k_m = 0.043\ Nm/A$, torque constant, named k;
- $k_e = 0.043\ Nm/A$, back emf constant, also called k;
- $J_m = 0.000056\ \mathrm{kg} * m^2$, that is inertia of the rotor;
- $J_l = 5\ \mathrm{kg} * m^2$, that is load inertia;
- $J = J_m + J_l * ((speed\_reducer)\hat{}2)$, that is inertia moment of the motor;
- $speed\_reducer = 1/100$, that is the value of gear reduction factor;
- $\tau_r = 0\ Nm$, that is the load torque;
- $b_m = 0.00081\ Nms/rad$, the rotational viscous friction coefficient;
- $b_l = 0\ Nms/rad$, that is the load viscous friction coefficient;
- $b = b_m + speed\_reducer^2 b_l\ Nms/\mathrm{rad}$, that is the friction coefficient;

Moreover, also the planar laser-cutting machine parameters have been specified, such as:

- $wheel\_radius = 0.1\pi$, of the wheel fixed on the motor axis;
- $\mathrm{max\_speed} = 1\ m/s$, that is the rotor velocity;
- $alpha = 1$, that is the speed reduction factor during any cutting operation;
- $cutting_{speed} = alpha * \max\_speed$, that is the machinery speed during the execution of cutting process;

The two axes DC motor controller has been shown below:

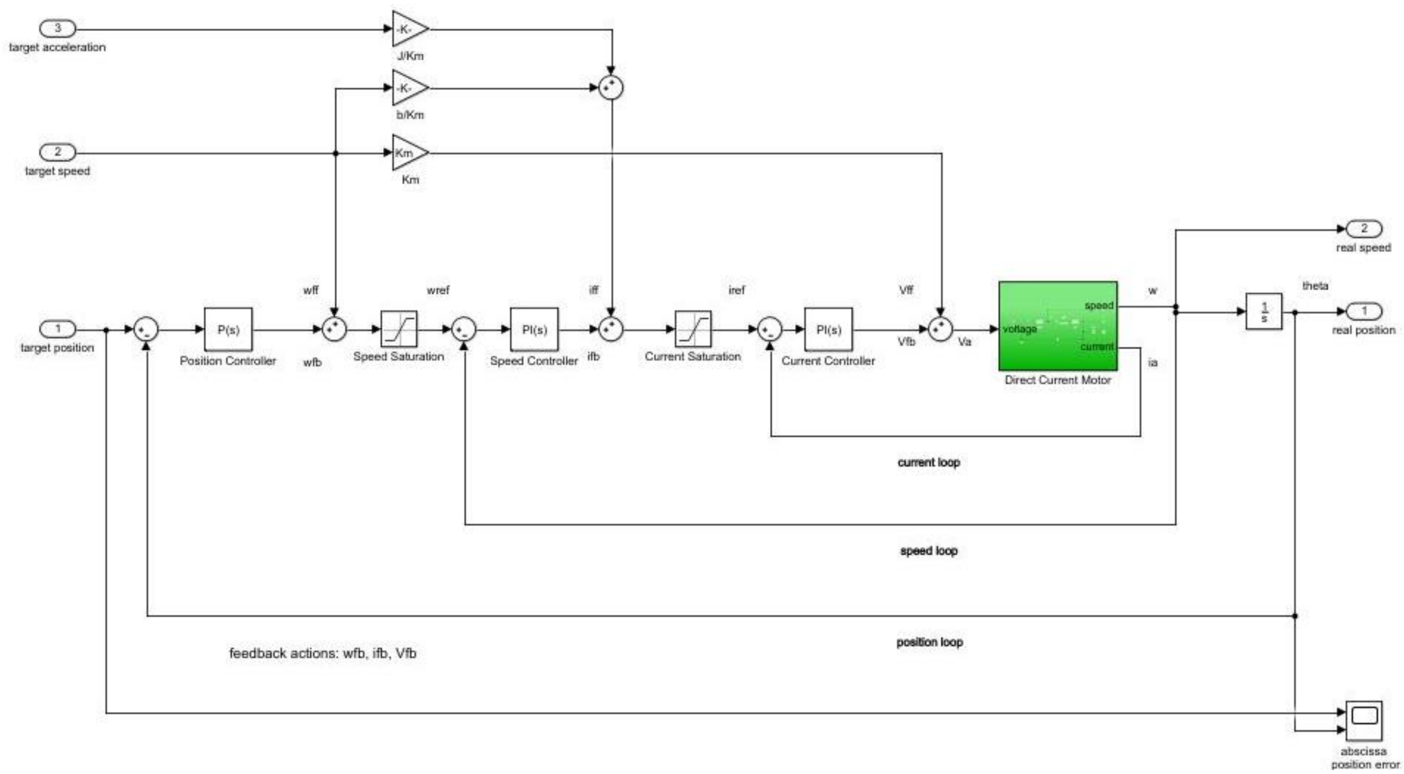


Figure 21: DC motor controller block diagram.

As mentioned above, PID blocks have been used to implement the control of the two-axes machinery. This strategy consists of a negative feedback system, known as “Proportional-Integral-Derivative” control and generally abbreviated as PID, that is very common in industrial applications. This kind of controller bases its working on the acquisition of a value as input from a specified process and then on the comparison of this with a reference value. The difference between the target value with the effective one is called error signal and it’s used to compute the value that the controller should produce as its output to correct the process variable. A PID controller regulates the output taking account of:

- the value of the error signal, if the PID implements a proportional action;
- the past values of the error signal, if the PID realizes an integral action;
- the time in which the error signal changes its value, if the PID makes a derivative action;

In comparison with more complex control algorithms, PID controllers are relatively simple to install and calibrate, because of the use of some empirical rules to do that, even if they aren’t capable to adapt their behaviour to the changes of process parameters. Generally, PID regulators are control algorithms that should be used in closed loop control systems, where the control input is given by the sum of three components, that are the same said before.

In mathematical terms, the expression in the time domain of the signal produced from a general PID controller is:

$$u(t) = k_P e(t) + k_I \int_{t_i}^{t_f} e(\tau)\, d\tau + k_D \frac{de(t)}{dt}$$

Where:

- $u(t)$ is the control signal of the transfer function of the process;
- $e(t)$ is the error signal, that represents the deviation between the desired output and the output which is detected at the instant of time t;
- K_P is the gain of the proportional action, associated with a P controller;
- K_I is the constant of the integral action, which referrers to an I controller;
- K_D is the constant of the derivative action, i.e. of a D controller;

So, the equation which describes a whole PID controller in Laplace domain is the following:

$$C_{PID} = k_P + \frac{k_I}{s} + k_D s$$

The three control actions can be implemented as a cascade of the corresponding three control blocks, as shown in the figure below.

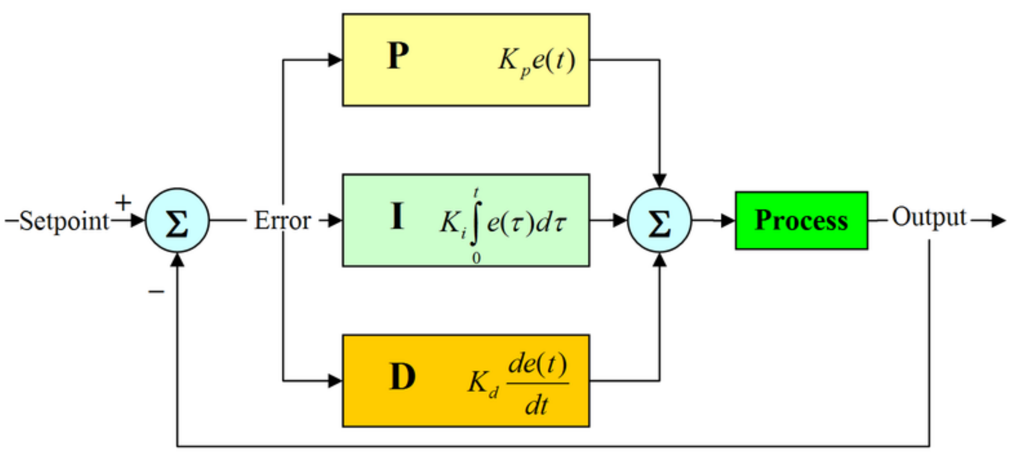


Figure 22: PID feedback control.

It’s specified that the setpoint value refers to $r(t)$ signal, that is the reference signal that the system receives as input.

The aim of this work is to determine the value of the voltage $V_a(t)$ to be applied to the system in order to realize the reference tracking of desired position, ensuring the stability of the system. So, the control of a direct current motor has been implemented considering that, in the absence of the counter-electromotive force, that has been managed by a feedforward action, the system can be considered as composed of a cascade of three blocks.

The control action has been realized separately for each of them, assuming the dynamic separation principle and as consequence, projecting each control loop considering negligible the dynamics of the others. So, the PID tuning has been done considering that each loop control is separated from the others and that the inner one should be faster than the external one.

Therefore, the PID constants have been chosen as follows:

- K_P =1, which is the P controller of the position loop;
- $K_P = 10,\ K_I = 30$, that are the constants of the PI controller of the velocity loop;
- $K_P = 100,\ K_I = 300$, which are the constants of the PI controller of the current loop;

These values are the same, both for the direct current motor controller along abscissa axis and for the one along ordinate axis.

The stability of the system has been verified, before considering what kind of contribution has been given by the introduction of the PIDs in it.

So, the Bode plots have been reported below:

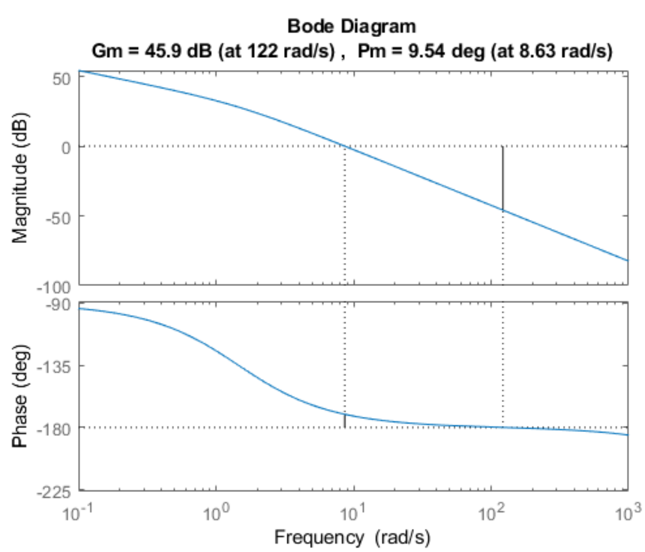
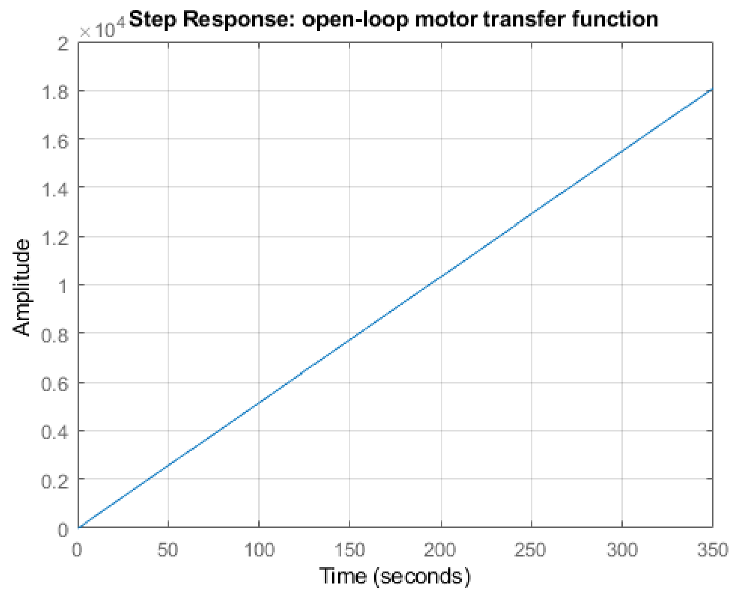


Figure 23: Bode plots of open loop motor transfer function.

As can be seen from the figure, the open-loop transfer function of the system in question is unstable.



Moreover, also the step response brings to the same conclusion.

The whole Simulink control scheme of the planar laser-cutting machine has been represented below:

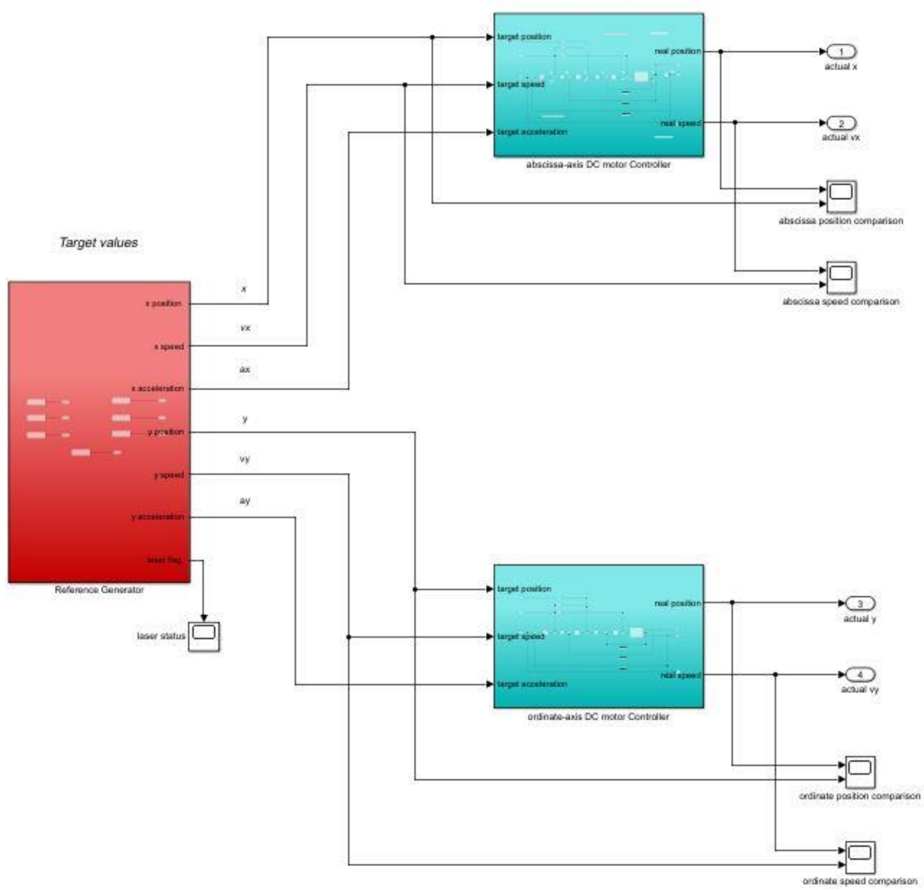


Figure 24: Planar laser-cutting machine control scheme.

After the implementation and the tuning of a specified controller for each control loop, the results that have been obtained about the whole system stability, shown as step-responses of the regulator closed-loop transfer function, are the following:

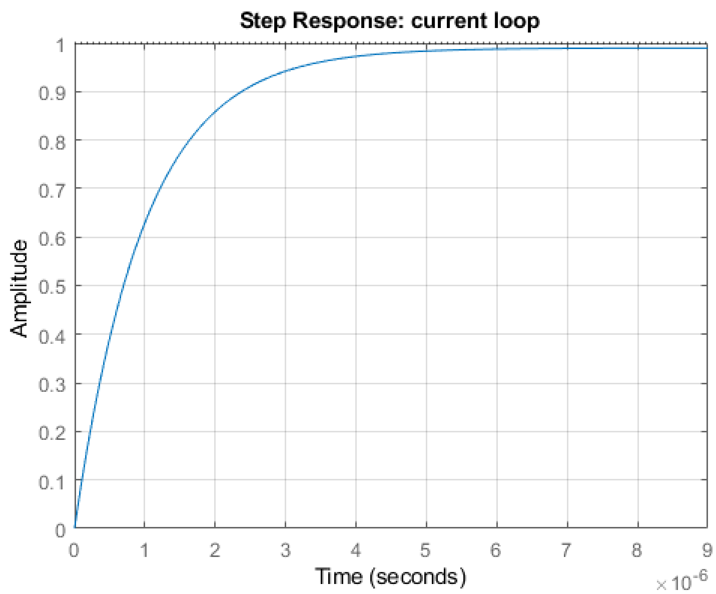


Figure 25: Step response of current loop using PI controller.

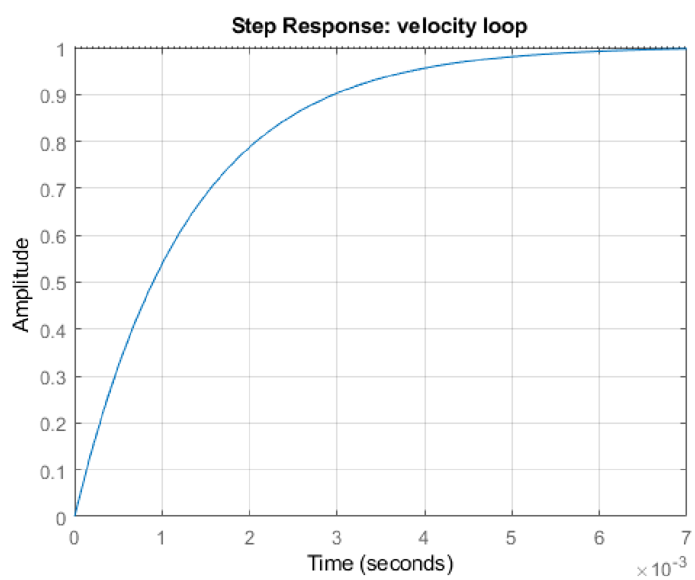


Figure 26: Step response of velocity loop using PI controller.

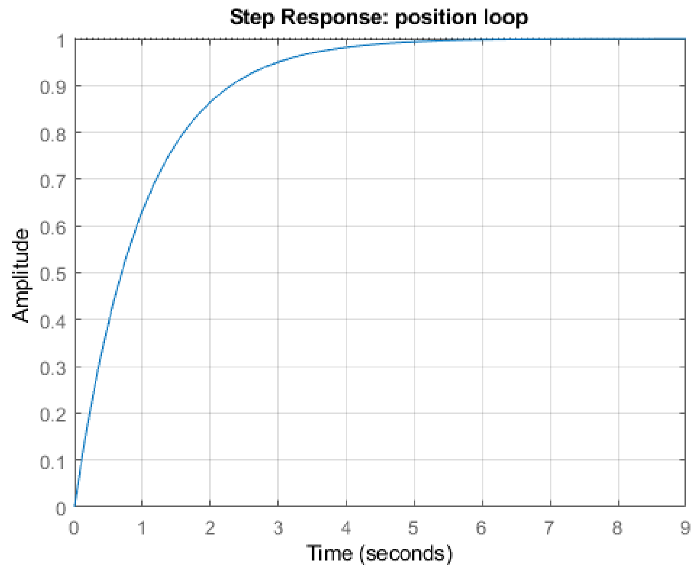


Figure 27: Step response of position loop using P controller.

After the addition of two PI controllers and one P regulator, the stability of the system has been guaranteed, as shown in the plot of the closed-loop system transfer function.

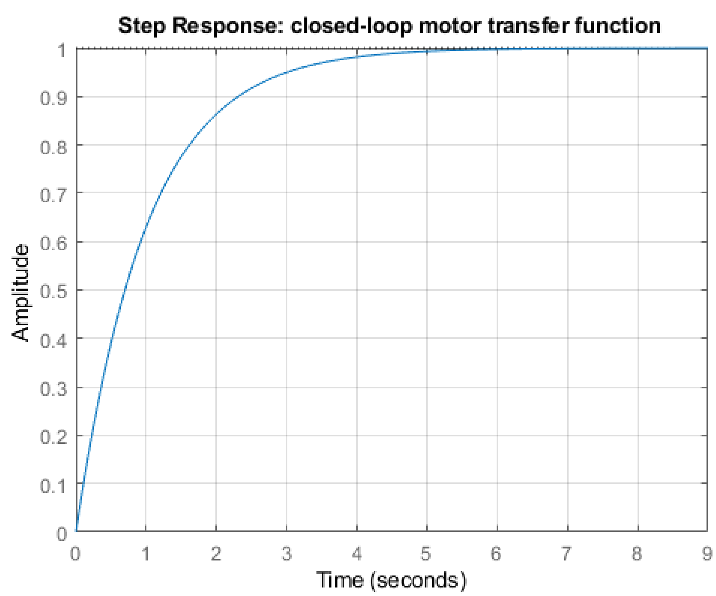


Figure 28: Step response of closed-loop system transfer function.

This is characterized by a Settling Time as 3.9175, a Rise Time as 2.2004, a Peak value equal to 1.0000 at the time of 10.5458 s and any overshoot and undershoot.

The effectiveness of the control action can be verified through the results shown by the Simulink scopes, which can be used to compare the desired trajectory with the one that has been really obtained.

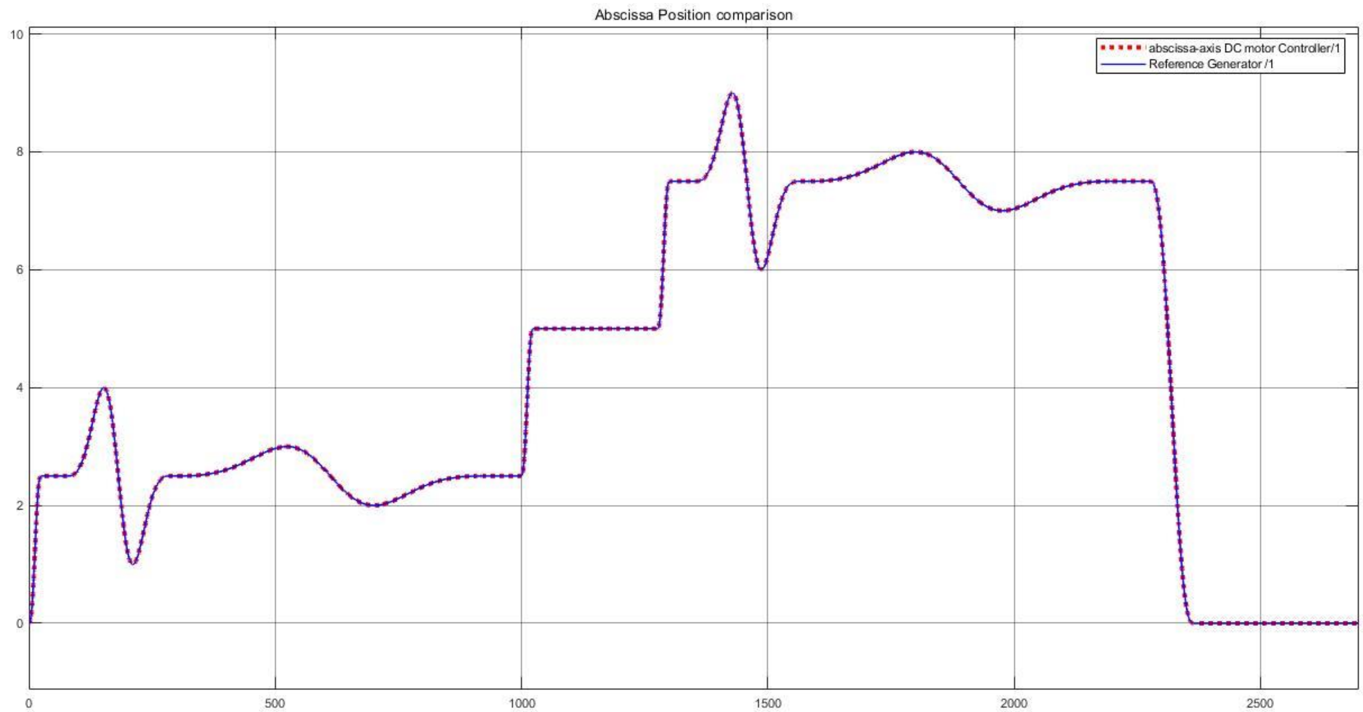


Figure 29: Comparison between desired and actual position along abscissa axis.

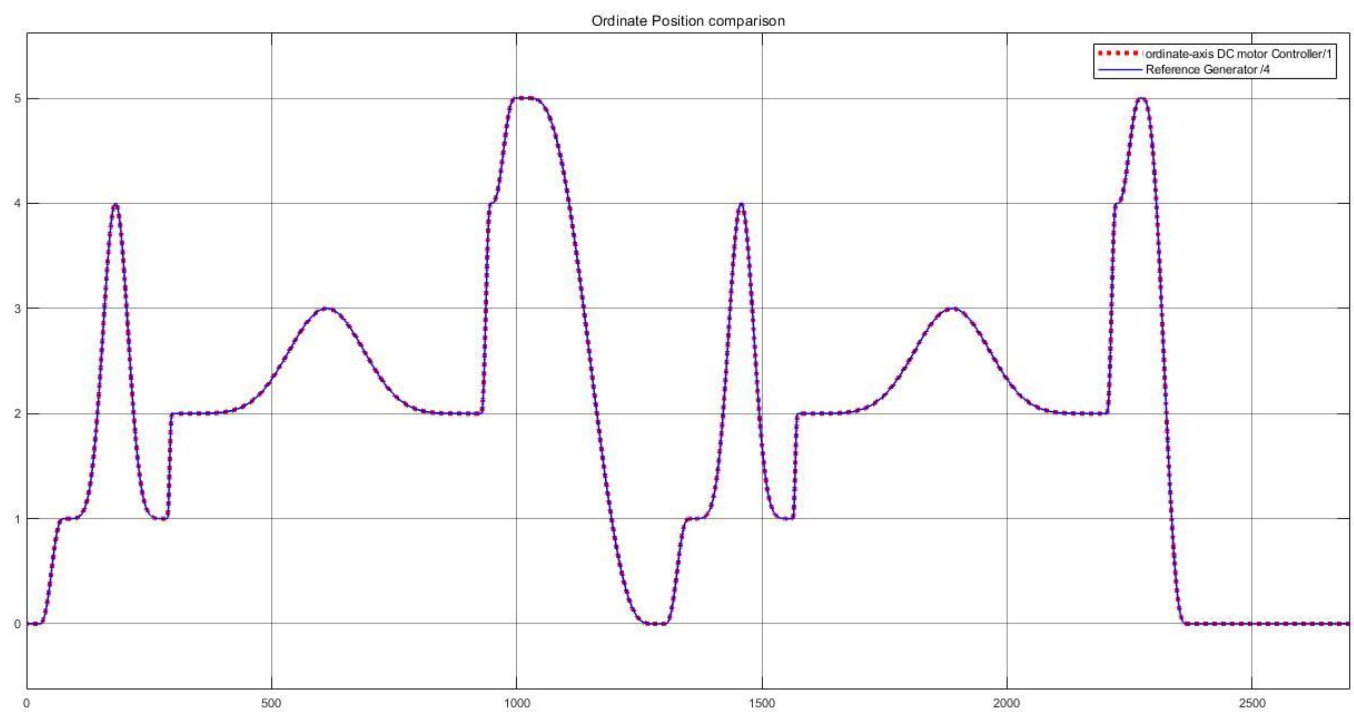


Figure 30: Comparison between desired and actual position along ordinate axis.

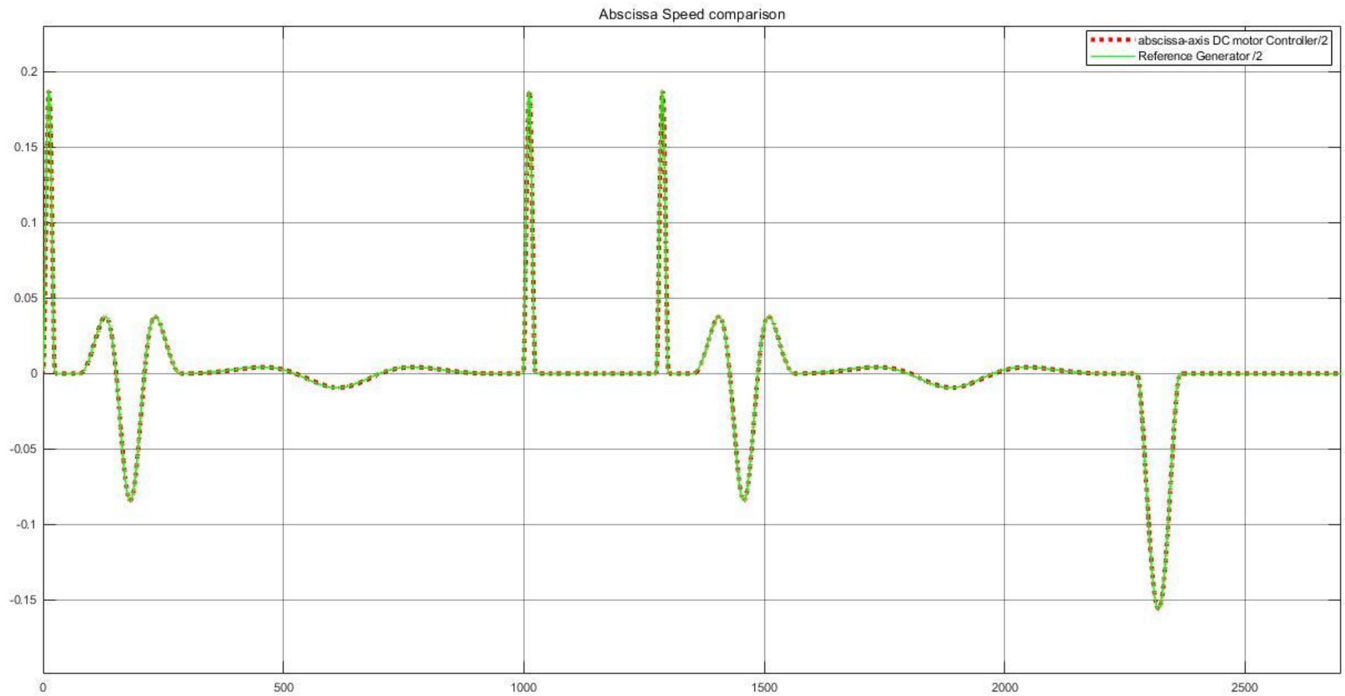


Figure 31: Comparison between desired and actual velocity along abscissa axis.

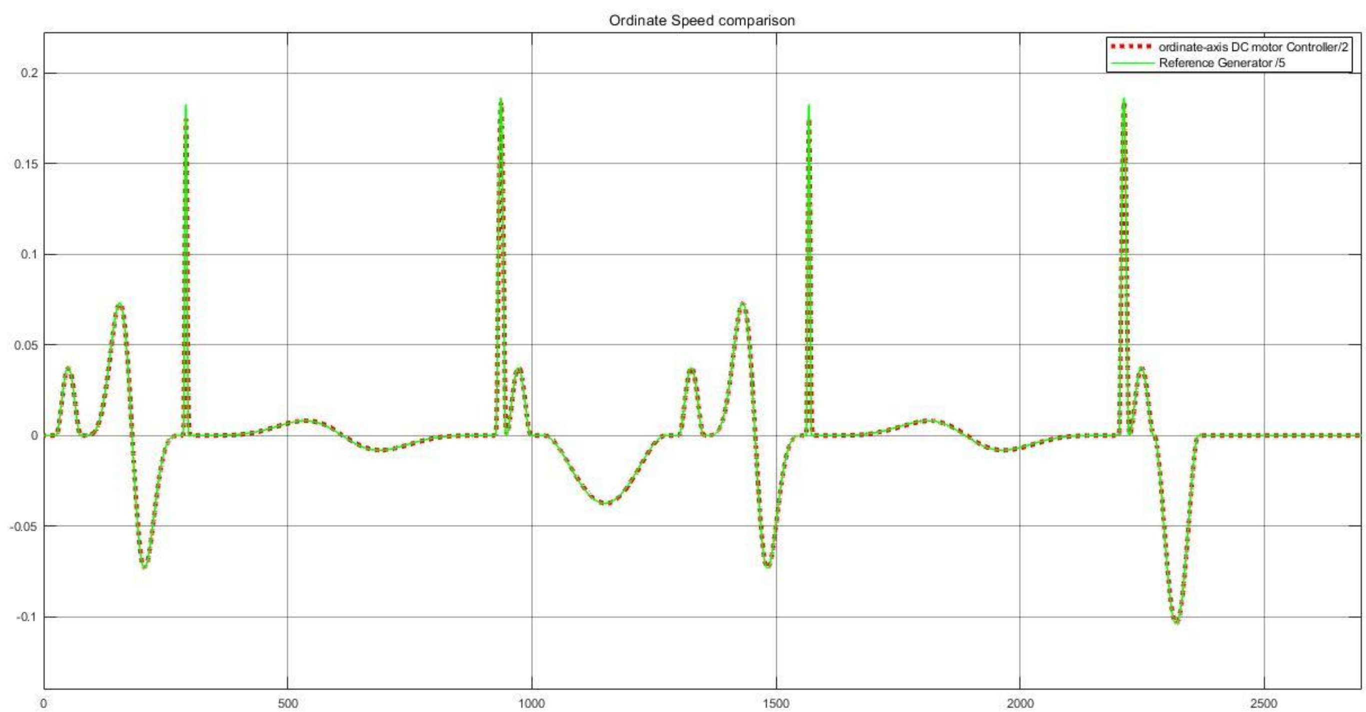


Figure 32: Comparison between desired and actual velocity along ordinate axis.

The position error has been also showed and it represents the deviation from the desired trajectory to the real one, which is the path that has been effectively tracked by the planar laser-cutting machine.

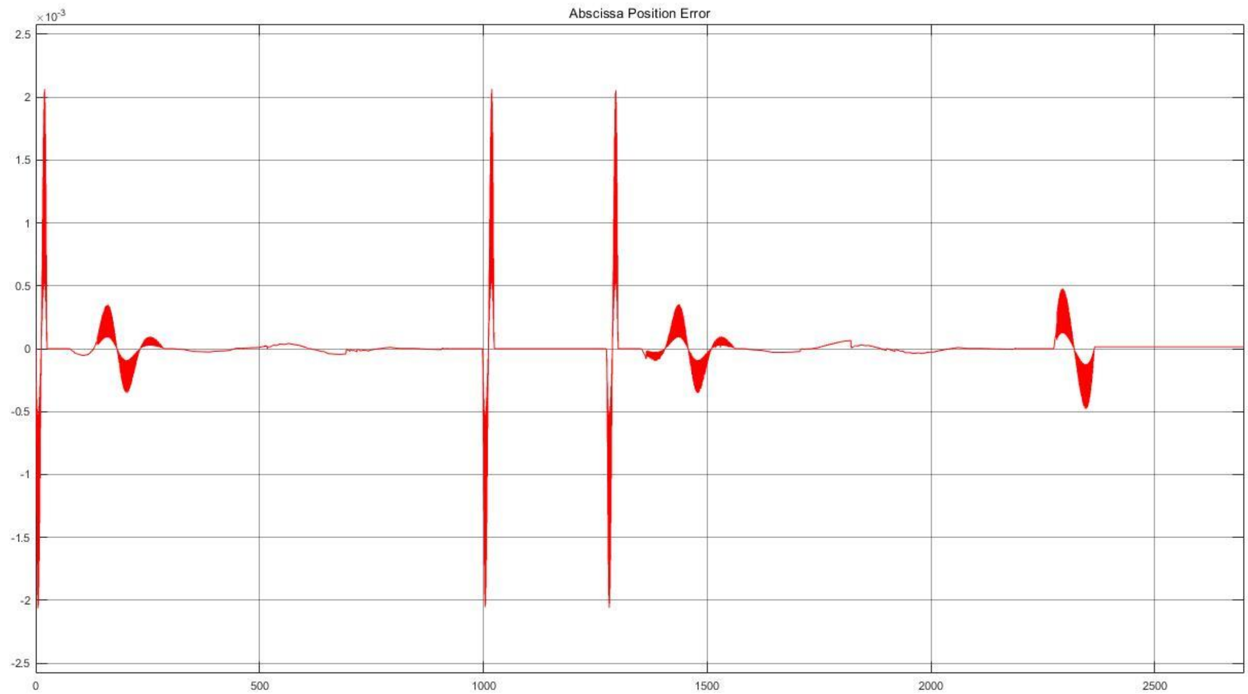


Figure 33: Error comparison between target and real position along x-axis.

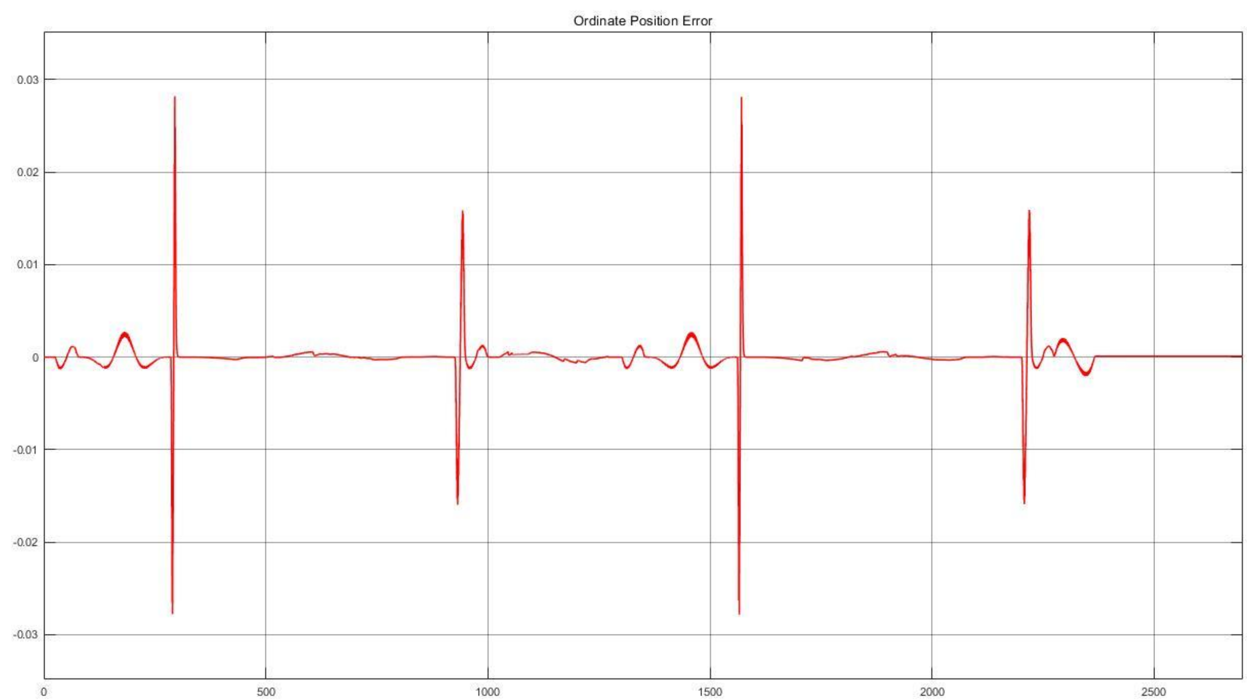


Figure 34: Error comparison between target and real position along y-axis.

As shown in figures above, the error that has been committed during the trajectory tracking is in the order of 10^{-3} values regarding x-axis and in the order of 10^{-2} for y-axis.

Chapter 2: Sequential Function Chart diagram

The second part of this report has been focused on the implementation of the plant for loading and unloading of goods by self-propelled carts.

First, an overview on PLCs and their several programming languages has been done.

2.1. *Programmable Logic Controller*

The term "Programmable Logic Controller", often in abbreviation "PLC", refers to an industry device used in the management or control of several industrial processes. This kind of computer works executing a program and processing digital and analogue signals that have been generated by sensors and then received by the actuators which belong to the industrial plant in question. Nowadays, these devices have been also utilized into domestic environment, thanks to the progressive miniaturization of their electronic components and their low cost.

A PLC is a modular hardware object, characterized by robustness to interference, high temperature and great humidity and it's capable to work, for a long time, on system that can never stop.

PLC structure depends on the process that needs to be automated, but it's generally composed of a power supply unit, of the CPU that can have internal or external RAM, ROM, EPROM or EEPROM memory, of several digital input and outputs cards, and if it's necessary also the analogue ones. If the PLC operates in a network with other PLCs, communication cards are needed too.

Generally, a logic controller can be defined as a device that relates input logical variables to output variables, using a set of combinational and / or sequential algorithms. The logic controller is defined as static, if the outputs depend on the values of the inputs at the same time, while it's said dynamic if equations that bind inputs and outputs are of the sequential type, so the outputs of the system also depend on the past values of the inputs.

This kind of controllers can be implemented by means of a set of physical devices that create logic gates, such as AND, OR, NOT and delay elements that define the sequential algorithm through their interconnection or by a programmable electronic system and a suitable stored control program. So, in the first case, it's called wired logic controller and in the other case it's named as programmable logic controller.

A logic or sequential controller is characterised to be easily programmable and reprogrammable in minimum interruption times, robust and therefore made with components and materials suitable for operation in an industrial environment, modular to permit maintenance and repairing actions. Moreover, it should have an easily expandable configuration, small dimensions and low energy consumption, the capability to interface with sensors, actuators and centralized systems in a simple way and an expandable internal memory for programs and data. Generally, the PLC is equipped with an operating system and several user-oriented programming modes, such as a graphic programming language that represents schematics of electric circuits.

In 1993, the international electrotechnical committee (IEC) has issued a standard about the hardware and software structure of this component, named as IEC 61131. In particular, this document defines the PLC as "an electronically operated digital system, intended for use in the industrial environment, which uses a programmable memory for the internal storage of user-oriented instructions for the implementation of specific functions, such as logic, sequencing, timing, counting and arithmetic calculation, and to control various types of machines and processes through both digital and analogue inputs and outputs ".

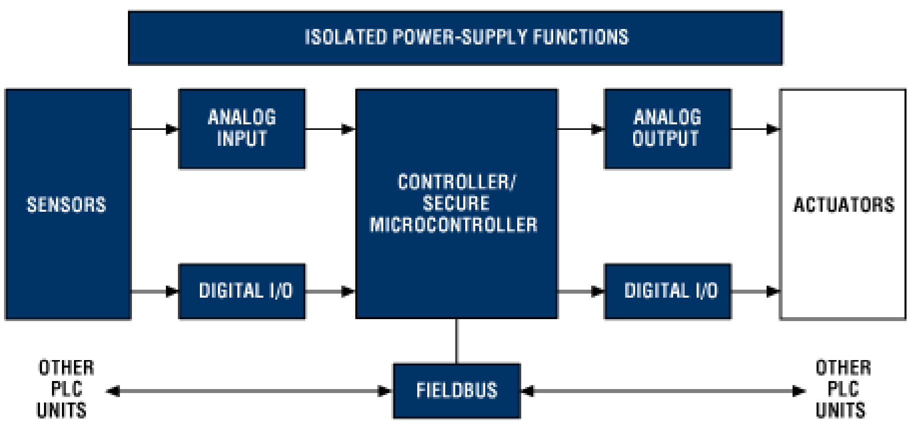


Figure 35: PLC scheme.

2.2. Programming languages

The PLCs should be programmed using a specialized software that runs on a PC, which allows to create programs to download in the PLC CPU memory.

In the IEC 61131 document, five programming languages have been standardized, which are three graphical programming languages, that are *Ladder diagram*, *Sequential function* chart and *Function Block Diagram* and two textual one, namely *Instruction List* and *Structured Text*.

2.2.1. Ladder Diagram

The Ladder Diagram, also called “contact language”, is the computer transposition of electrical circuits that have been used in the definition of sequential wire controllers. So, logic symbols, such as AND, OR and NOT, corresponding to input and output signals are used to implement the control logic by drawing the electrical diagrams on the programming software and no longer by wiring the relays.

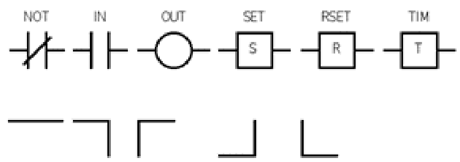


Figure 36: Ladder Diagram symbols.

So, this language is a graphical representation of electronic devices that realize logical expressions in which the operands are open or closed relays and their result is stored inside a coil. In fact, two vertical lines represent the power supply and delimit the electrical components of the circuit and so the logical operations. Then, these sequential expressions are represented by horizontal lines that power the coils allowing the flow of the current, from the left line to the right one. Contacts can allow or deny the passage of current from left to right, depending on their state, realizing the behaviour of a Boolean variable. In the same way, coils can be powered or not, thus storing the true or false value of the circuit connected to them.

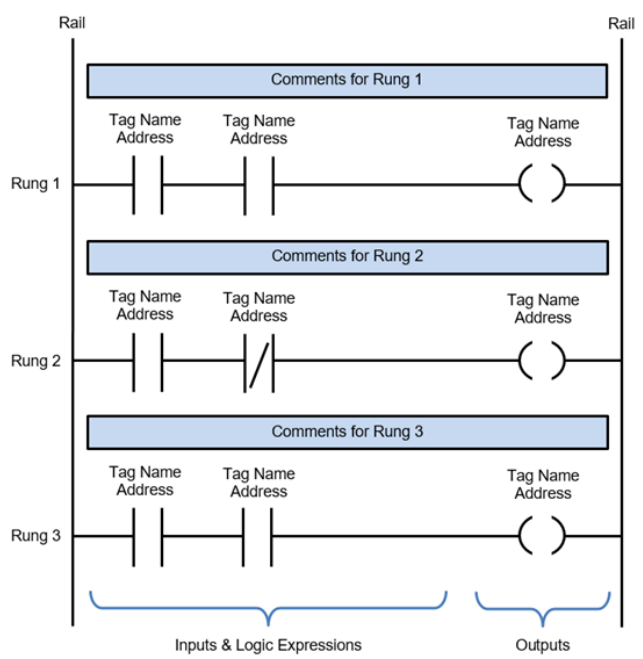


Figure 37: Example of a LD structure.

Ladder programming follows two fundamental principles, which are that the energy flow moves from left to right and the execution of instructions always takes place from top to bottom. Moreover, contacts let the current flow only in the presence of a rising edge of the variable associated with them, in fact, the current can flow though one of them only if its associated variable takes the true value, after that it's has been characterized by the false value in the previous cycle. These contacts are graphically characterized by the letter P and there are also some contacts that let the current flow in the presence of a falling edge of the associated variable, which are indicated by the letter N.

2.2.2. Sequential function chart

The sequential function chart is a graphic programming language used for modelling the sequential evolution of an automation system, by implementing a finite state automaton, whose machine cycle is represented as a series of sequential control actions. As it's a high-level programming language, it can be considered a graphic formalism for the description of logical actions, in fact, in the implementation of a sequential control program, actions and conditions should be better specified, and so other languages, such as Ladder Diagram, have been necessary. The SFC is composed of three fundamental graphical elements, that are state, transition and directed arc.

The *state* or step is a situation of the system that can change or be modified only at the occurrence of a certain event. So, it's a specific operating condition of the system in question which refers to determined actions that should be accomplished. Then, the evolution of the system from one state to another occurs when the corresponding events happen. So, a state can be active, or inactive, and to it has been associated several control instructions that will be executed only when the status is active.

Generally, states are represented by rectangles in which the representative and unique name of the state is inserted, such as numerical element. Moreover, the actions which are associated to each state are usually figured as a rectangle too, in which these have been described and it's connected to the state which it refers.

Transitions represent the possibility of evolution from a state to another, when a condition that has been associated to the transition in question has been verified. These transitions, usually expressed through Boolean functions and represented as a horizontal line that cuts the arc that unites two states, are used to describe the events that could change the operating status of the system. Finally, the evolution from one state to another is univocally determined by the directed arc that unites them and by the transition, in addition to the relative condition, which determines the possibility of this evolution to happen.

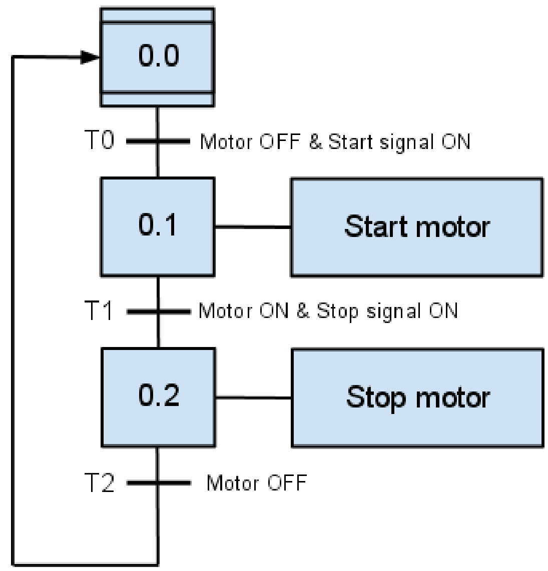


Figure 38: Example of SFC program.

So, the sequential control via an SFC diagram gives the possibility to describe, in a simple and unambiguous way, the specifications of sequential operation of an automatic system, to overcome the limits of finite state automata, for example by allowing the parallelism.

2.3. Self-propelled Carts Project

The automation process that should be implemented is a system for loading and unloading of goods by means of two self-propelled carts. Their movement takes place along two rail lines, that join into one where two carts share the same area. These zones are the rest area and the unloading one. On the contrary, the loading area and the waiting one are separated, in fact they consist of two different tracks, on which each cart can move itself back and forth, according to the tasks that it should accomplish. The access on the shared areas has been managed by a semaphore that guarantees the mutual exclusion of the shared track. So, it has the aim to give a signal to an actuator which should activate the rail switch of this common resource in favour of A cart or B cart, so any collision can occur. Moreover, both in proximity of the loading area and the unloading one, some actuators are present to realize loading and unloading operations of goods.

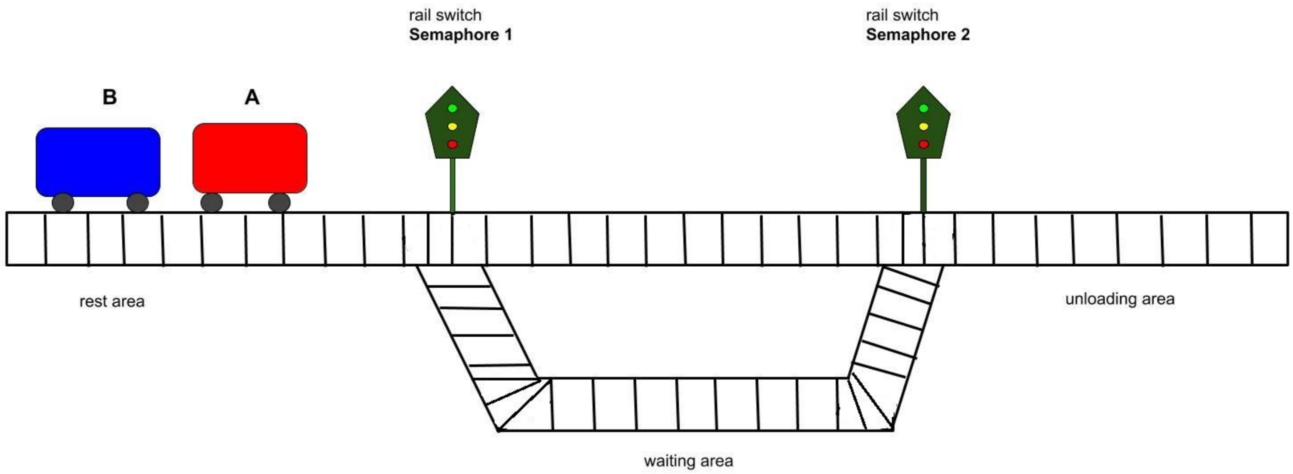


Figure 39: Self-propelled cart system.

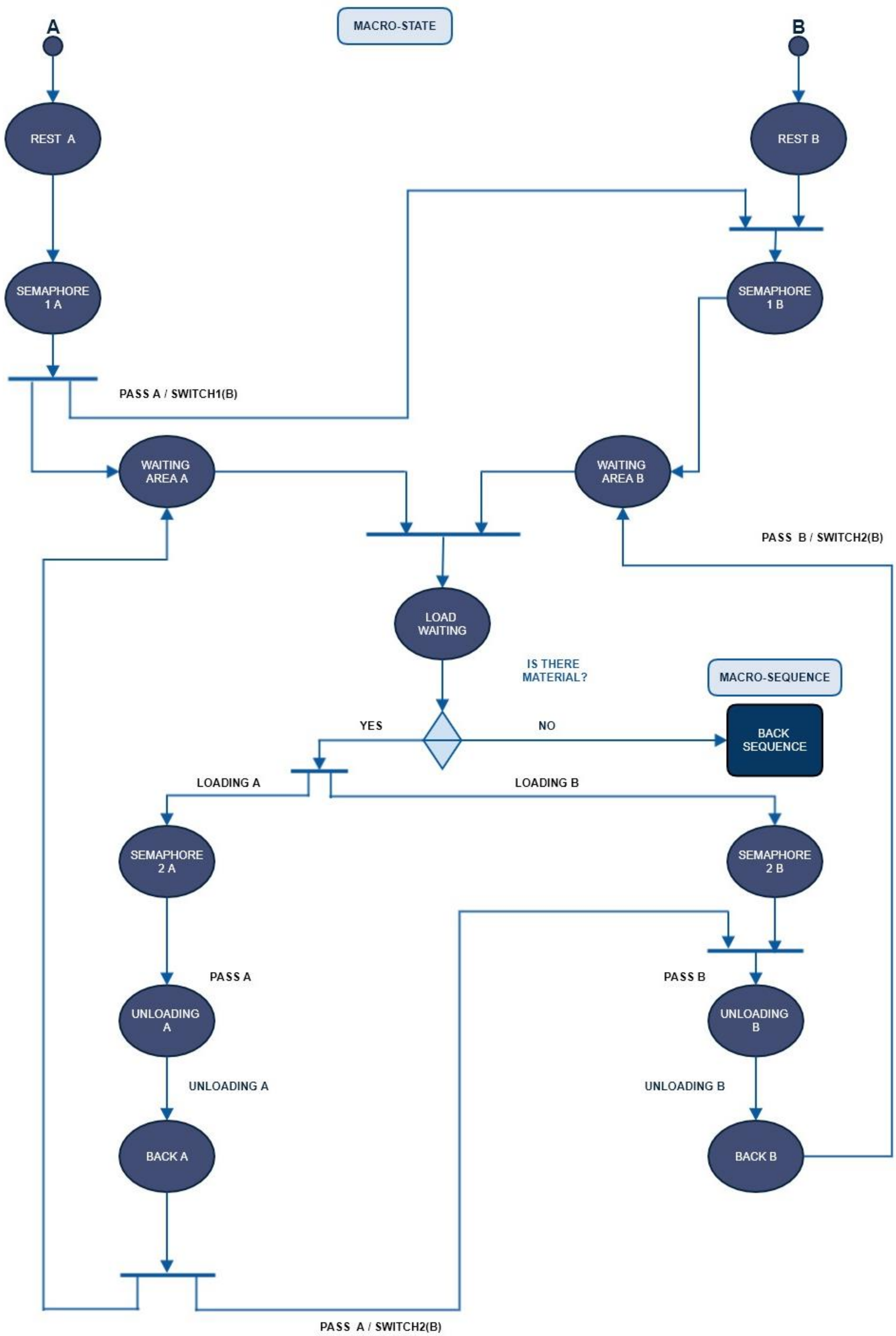
The project purpose is to solve this problem making the carts move themselves independently from each other, ensuring the absence of crashes in the shared zone.

The initial position of the carts consists in the rest area and it's important that their disposition on it respects that A cart follows B cart. So, A cart is always the first to start, when the whole automation system has been activated by an appropriate signal. Then, the presence of the two carts on the loading and unloading area will be verified, before starting the related operations, thanks to the use of specified sensors.

2.3.1. Finite-state machine

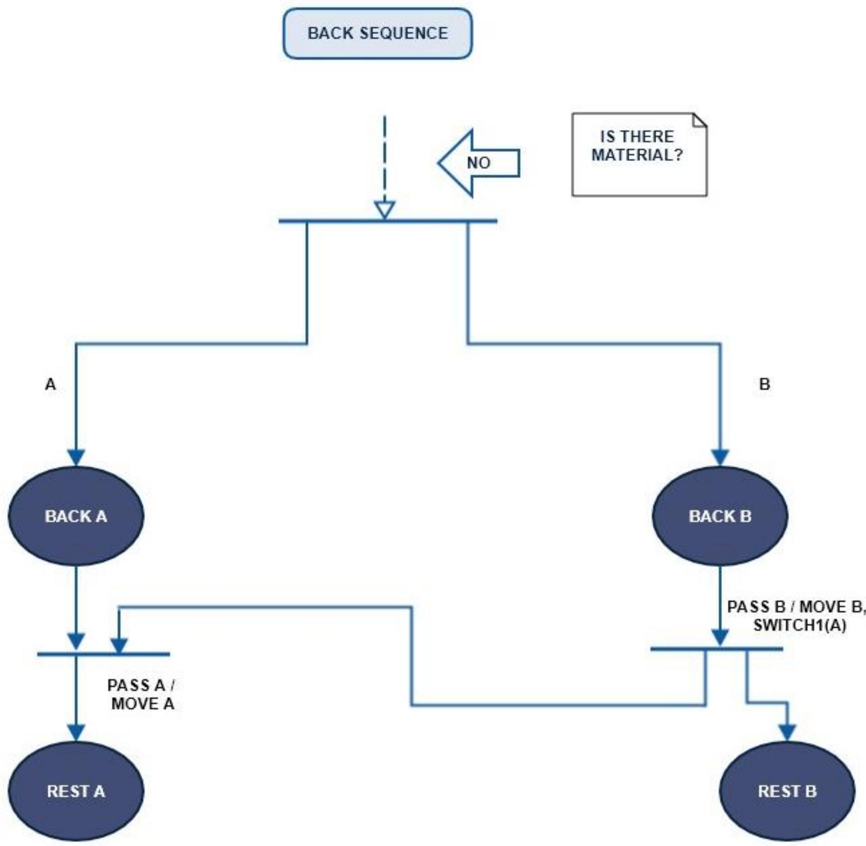
A finite-state machine or finite-state automaton is a mathematical model of computation, an abstract machine that is composed of a finite number of states. It can change from one state to another in response of some external inputs, that can verify or deny determined conditions that have been assigned to a specified transition. So, a finite-state machine is defined by a list of its states, its initial state and the condition for each transition.

Now, a representation of a finite-state machine, describing the automation system which has been reported in this work, has been figured as follows:



A
MACRO-STATE
B
REST A
REST B
SEMAPHORE 1 A
SEMAPHORE 1 B
PASS A / SWITCH1(B)
WAITING AREA A
WAITING AREA B
PASS B / SWITCH2(B)
LOAD WAITING
IS THERE MATERIAL?
MACRO-SEQUENCE
YES
NO
BACK SEQUENCE
LOADING A
LOADING B
SEMAPHORE 2 A
SEMAPHORE 2 B
PASS A
PASS B
UNLOADING A
UNLOADING B
UNLOADING A
UNLOADING B
BACK A
BACK B
PASS A / SWITCH2(B)

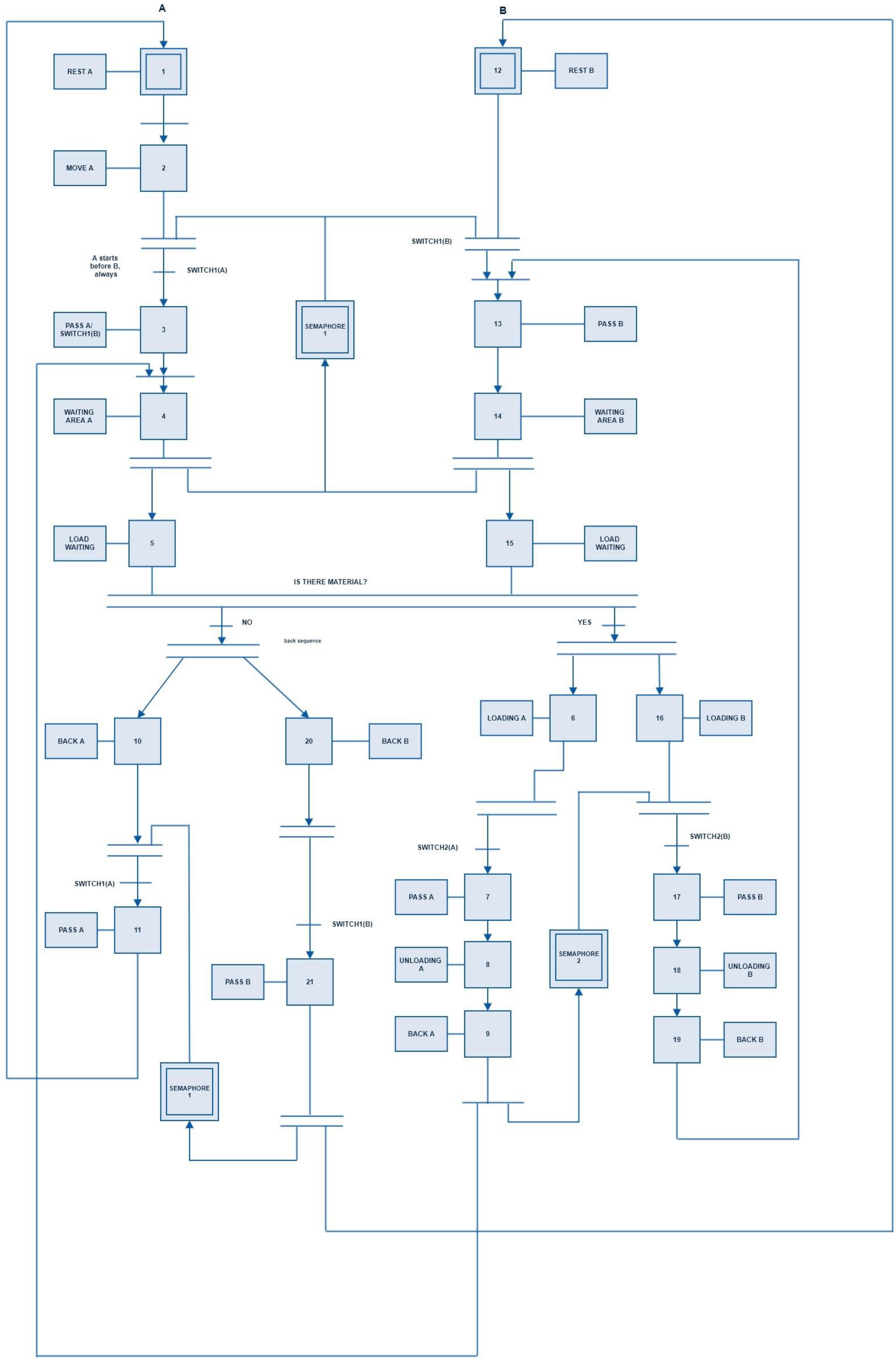
The explication of the macro-sequence called "Back sequence" has been shown below:



To do this finite-state automaton the Mealy machine has been used. It's a finite-state automaton whose output values are determined by the current state and the current input, unlike the Moore machine, which instead works only as a function of the current state. In this case, an action has been associated to the arc transition and any possible transition can specify a different action.

2.3.2. SFC diagram implementation

Finally, as required by project specifications, the sequential control of the industrial process described above has been realized. For its implementation, the graphical programming language called SFC has been used to draw the following diagram:


A
B
REST A
1
12
REST B
MOVE A
2
SWITCH1(B)
A starts before B, always
SWITCH1(A)
PASS A/ SWITCH1(B)
3
SEMAPHORE 1
13
PASS B
WAITING AREA A
4
14
WAITING AREA B
LOAD WAITING
5
15
LOAD WAITING
IS THERE MATERIAL?
NO
YES
back sequence
LOADING A
6
16
LOADING B
BACK A
10
20
BACK B
SWITCH2(A)
SWITCH2(B)
SWITCH1(A)
SWITCH1(B)
PASS A
7
17
PASS B
PASS A
11
UNLOADING A
8
SEMAPHORE 2
18
UNLOADING B
PASS B
21
BACK A
9
19
BACK B
SEMAPHORE 1

Conclusion

In this report, two problems related to industrial automation systems have been considered.

In the first part of this work, a planar laser-cutting machine, which it's characterized by a control-axes scheme, has been designed to realize several assigned shapes from a laminated sheet, which is the material that should be worked to obtain a finished product. For this reason, a control action on the two equipped DC motors has been implemented, to make them follow the trajectory which describes the figures to cut from the metal sheet. So, the control problem has been focused on the path planning through polynomial functions called spline, ensuring position, velocity and acceleration constrains, respecting however motor physical limits. The control action has been implemented using MATLAB environment and then Simulink software, in fact PID controllers have been used to guarantee system performance.

In the second part, an automated process of loading and unloading of goods, through two carts moving on rail tracks, has been considered. Generally, this kind of problems are related to PLC applications in an industrial environment. So, to formalize it, a graphical programming language that is Sequential Function Chart has been chosen and the corresponding diagram has been drawn.