Department of CONTROL AND COMPUTER ENGINEERING (DAUIN)

Master’s degree in Mechatronic Engineering

A.A. 2020/2021

Robotics – Prof. Rizzo and Prof. Primatesta

Robotics Optional Project

Anthropomorphic robotic arm for drawing applications

Federica Parisi – 289819

Martina Bonaffini – 289563

Nicola Occhipinti – 289648

Ronald Cristian Dutu – 290185

Introduction

This project is based upon the building of a robotic arm, in particular an anthropomorphic robot without the spherical wrist, implemented in an open chain fashion with four links and three revolute joints. We would like to control the arm in such in order to draw simple letters on a sheet paper using a pen attach to the end part of the manipulator. Furthermore, we would like to investigate features regarding direct and inverse kinematics, singularities and the study of trajectory that the robot has to follow to accomplish the task. We mainly used the Robotic Toolbox by Peter Corke and MATLAB Support Package for Arduino Hardware by MathWorks.

Figure 1: Anthropomorphic robot arm after being assembled

HARDWARE Set up

The hardware parts of the robotic arm have been produced by Adeept. There are two configurations of the arm: one that uses three joints and one that uses 5 joints. We decide to build and analyse the configuration with only three servo motors because it is the most appropriate to satisfy our task, that is to write some letters decided by us in a paper sheet. The hardware component that we use are:

- **Adeept arm drive board**

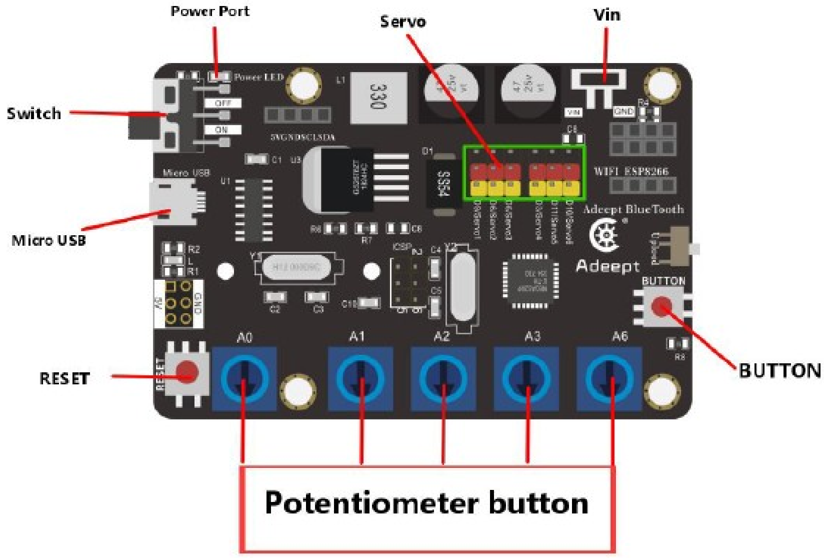

Figure 2: *Adeept arm drive board with a description of all the components inside it*

It is similar to the Arduino UNO development board, including the hardware part and the software part. The board is mainly composed of a microcontroller, a universal I/O interface, a potentiometer able to change the resistances of the servo in order to control the movement of the robotic arm, etc.

- **Three servo motors**

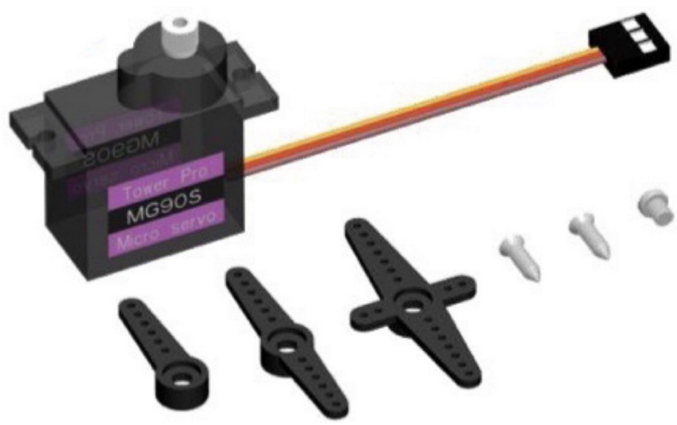

Figure 3: *Servo motor used in the robotic arm*

This is an automatic control system that enables the object's position, orientation, state and other output-controlled quantities to follow arbitrary changes in the input target. The servo mainly depends on pulse for location: it means that the servo motors receive an input and rotates the angle corresponding to the impulse to realize displacement. Because the servo motor itself has the function of sending out pulses, the servo motor rotates every time at an angle, and corresponding number of pulses will be sent out. In this way, the system will know how many pulses are sent to the servo motor and how many pulses are received in such a way that it is possible to control the rotation of the motor, achieving precise positioning.

MATLAB Set up

Before proceeding with the actual robot calculations, we first of all developed MATLAB code to be able to see and experiment a model of the robot in this environment.

Which then gives the following graph:

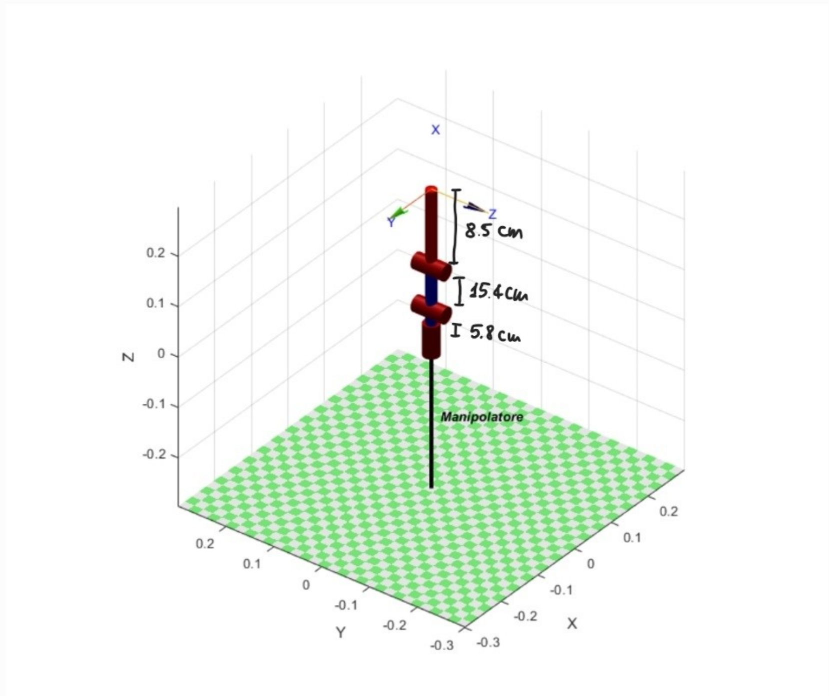

Figure 4: Robot representation + measurements

You can see that we also highlighted the measurement of each link, so in our model we considered the actual measurement of the real robot.

Kinematics

The Robot Kinematics is the study of the geometry of the arm motion with respect to a reference frame that could be centred on the base of the robot or could be referred to another point on the space. There are two different kinematics study to make:

- **Direct kinematics:** since we have three servomotors and four links, the number of DOF correspond to three and this is the number of joints variables to consider. We start the analysis by placing each one of the four reference frames according to the DH convention.

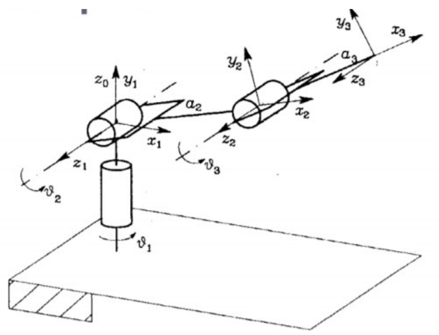

Figure 5: *Schematic of the anthropomorphic manipulator*

Before the computation of the matrix, we have to build the table containing all the parameters needed to simplify the calculus of the desired matrix exploiting the Denavit-Hartenberg structure that is reported below.

$$T_3^0(\theta_1,\theta_2,\theta_3)=A_1^0(\theta_1)\,A_2^1(\theta_2)\,A_3^2(\theta_3)$$

Before the computation of the matrix, we have to build the table containing all the parameters needed to simplify the calculus of the desired matrix exploiting the Denavit-Hartenberg structure that is reported below.

$$A_i^{i-1}(q_i)=A_{i'}^{i-1}\,A_i^{i'}=\begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where:

- a_i is the distance between O_i and $O_{i'}$
- α_i is the angle between z_{i-1} and z_i about x_i
- d_i is the coordinate of $O_{i'}$ along axis z_{i-1}
- θ_i is the angle between x_{i-1} and x_i about z_{i-1}

Link	**a_i**	**α_i**	**d_i**	**θ_i**
1	0	π/2	1	θ_1
2	a_2	0	0	θ_2
3	a_3	0	0	θ_3

Table 1: *schematic of the anthropomorphic manipulator*

The translation of such a table in MATLAB is the following one:

After defining the DH parameters, we can now write the intermediate transformation matrices and then compute the final multiplication to get the transformation matrix that links the end effector frame 3 with the base frame 0.

$$A_1^0(\theta_1)=\begin{bmatrix} c_{\theta_1} & 0 & s_{\theta_1} & 0 \\ s_{\theta_1} & 0 & -c_{\theta_1} & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2^1(\theta_2)=\begin{bmatrix} c_{\theta_2} & -s_{\theta_2} & 0 & a_2 c_{\theta_2} \\ s_{\theta_2} & c_{\theta_2} & 0 & a_2 s_{\theta_2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2(\theta_3)=\begin{bmatrix} c_{\theta_3} & -s_{\theta_3} & 0 & a_3 c_{\theta_3} \\ s_{\theta_3} & c_{\theta_3} & 0 & a_3 s_{\theta_3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2^0(\theta_1,\theta_2)=A_1^0(\theta_1)\,A_2^1(\theta_2)=\begin{bmatrix} c_{\theta_1} c_{\theta_2} & -c_{\theta_1} s_{\theta_2} & s_{\theta_1} & a_2 c_{\theta_1} c_{\theta_2} \\ s_{\theta_1} c_{\theta_2} & -s_{\theta_1} s_{\theta_2} & -c_{\theta_1} & a_2 s_{\theta_1} c_{\theta_2} \\ s_{\theta_2} & c_{\theta_2} & 0 & a_2 s_{\theta_2}+d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^0(\theta_1,\theta_2,\theta_3)=A_2^0(\theta_1,\theta_2)\,A_3^2(\theta_3)=¿\begin{bmatrix} c_{\theta_1} c_{\theta_2} c_{\theta_3}-c_{\theta_1} s_{\theta_2} s_{\theta_3} & -c_{\theta_1} c_{\theta_2} s_{\theta_3}-c_{\theta_1} s_{\theta_2} c_{\theta_3} & s_{\theta_1} & c_{\theta_1} c_{\theta_2} a_3 c_{\theta_3}-c_{\theta_1} s_{\theta_2} a_3 s_{\theta_3}+a_2 c_{\theta_2} \\ s_{\theta_1} c_{\theta_2} c_{\theta_3}-s_{\theta_1} s_{\theta_2} s_{\theta_3} & -s_{\theta_1} c_{\theta_2} s_{\theta_3}-s_{\theta_1} s_{\theta_2} c_{\theta_3} & -c_{\theta_1} & s_{\theta_1} c_{\theta_2} a_3 c_{\theta_3}-s_{\theta_1} s_{\theta_2} a_3 s_{\theta_3}+a_2 s_{\theta_1} \\ s_{\theta_2} c_{\theta_3}+c_{\theta_2} s_{\theta_3} & -s_2 s_3+c_2 c_3 & 0 & s_{\theta_2} a_3 c_{\theta_3}+c_{\theta_2} a_3 s_{\theta_3}+a_2 s_{\theta_2}+d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **Inverse kinematics:**

The inverse kinematics problem consists of finding the joint variables corresponding to a given end-effector position and orientation.

Let $\mathbf{p_w}$=[p_{wx} p_{wy} p_{wz}] be the position of the wrist with respect to the base frame. Where:

$$p_{wx}=c_1(a_2 c_2+a_3 c_{23})$$
$$p_{wy}=s_1(a_2 c_2+a_3 c_{23})$$
$$p_{wz}=a_2 s_2+a_3 c_{23}$$

It is worth squaring and summing and yielding:

$$p_{wx}^2+p_{wy}^2+p_{wz}^2=a_2^2+a_3^2+2a_2 a_3 c_3$$

From which:

$$c_3 = \frac{p_{wx}^2 + p_{wy}^2 + p_{wz}^2 - a_2^2 - a_3^2}{2 a_2 a_3}$$

where the admissibility of the solution obviously requires that -1≤c_3≤1, otherwise the wrist
point is outside the reachable workspace of the manipulator. Hence it is:

$$s_3 = \pm\sqrt{1 - c_3^2}$$

and thus

$$\vartheta_3 = Atan2(s_3, c_3)$$

Having determined ϑ_3, it is possible to compute ϑ_2 as follows, squaring and summing:

$$p_{Wx}^2 + p_{Wy}^2 = (a_2 c_2 + a_3 c_{23})^2$$

Where:

$$a_2 c_2 + a_3 c_{23} = \pm\sqrt{p_{Wx}^2 + p_{Wy}^2}$$

The system of the equations, admits the solution:

$$c_2 = \frac{\pm\sqrt{p_{wx}^2 + p_{Wy}^2}(a_2 + a_3 c_3) + p_{Wz} a_3 s_3}{a_2^2 + a_3^2 + 2 a_2 a_3 c_3}$$

$$s_2 = \frac{p_{Wz}(a_2 + a_3 c_3) \mp \sqrt{p_{Wx}^2 + p_{Wz}^2} a_3 s_3}{a_2^2 + a_3^2 + 2 a_2 a_3 c_3}$$

Now, we can compute ϑ_2 and ϑ_1:

$$\vartheta_2 = Atan2(s_2, c_2) \qquad \vartheta_1 = Atan2(s_1, c_1)$$

Notice finally how it is possible to find the solutions only if at least:

$$p_{Wx} \neq 0 \quad \text{or} \quad p_{Wy} \neq 0$$

In the case $p_{Wx} = p_{Wy} = 0$, an infinity of solutions is obtained since it is possible to determine the joint variables ϑ_2 and ϑ_3 independently of the value of ϑ_1.

Singularities' Study

As we already mentioned multiple times, the robot is a 3 DOFs that can be considered as an anthropomorphic manipulator without the spherical wrist, therefore it inherits all the characteristics, this also means singularities. In this typology of structure, we can identify 2 common singularities sources: elbow totally stretched or retracted singularity and shoulder singularity. To understand what the singularity configuration are we have to compute the determinant of the Jacobian and impose that: det(J) = 0.

The problem is that the manipulator used is underactuated, meaning that it contains a number of joints which is less that the number of variables needed to consider every possible motion in space, which is 6. When computing the Jacobian, only three rows out of 6 are independent from each other. Therefore, we will consider just the first 3 rows of the Jacobian that gave us the relationship between joint velocities and linear end- effector velocity. So the actual used Jacobian used is:

$$J_P = \begin{bmatrix} -s_{\theta_1}(a_2 c_{\theta_2} + a_3 c_{\theta_2+\theta_3}) & -c_{\theta_1}(a_2 s_{\theta_2} + a_3 s \text{¿¿} \theta_2 + \theta_3) \text{¿} - a_3 c_{\theta_1} s_{\theta_2+\theta_3} \\ c_{\theta_1}(a_2 c_{\theta_2} + a_3 c_{\theta_2+\theta_3}) & -s_{\theta_1}(a_2 s_{\theta_2} + a_3 s \text{¿¿} \theta_2 + \theta_3) \text{¿} - a_3 s_{\theta_1} s_{\theta_2+\theta_3} a_3 c_{\theta_2+\theta_3} \text{¿} \\ 0 & \text{¿} \end{bmatrix}$$

The condition of singularities is realized when the determinant of the manipulator goes to zero.

$$det(J_P) = -a_2 a_3 s_{\theta_3}(a_2 c_{\theta_2} + a_3 c_{\theta_2+\theta_3}) = 0$$

There are two cases when this happens:

- $s_{\theta_3} = 0 \wedge so\,\theta_3 = 0 \vee \theta_3 = \pi$

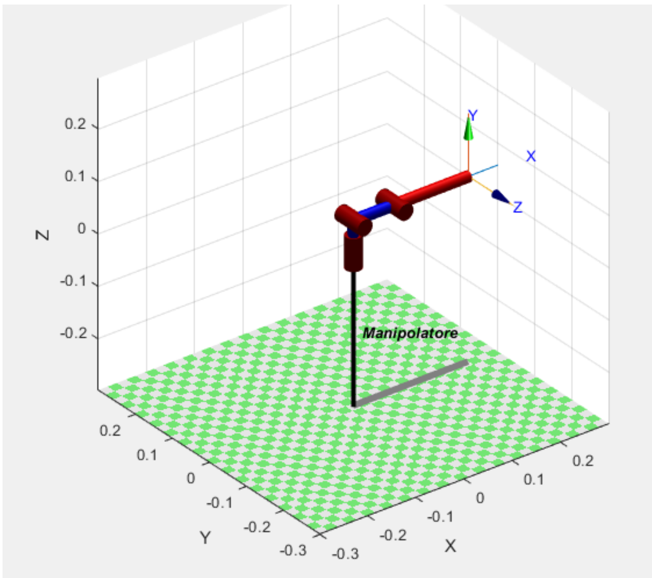

Figure 6: Elbow Singularity when $\theta_3 = 0$

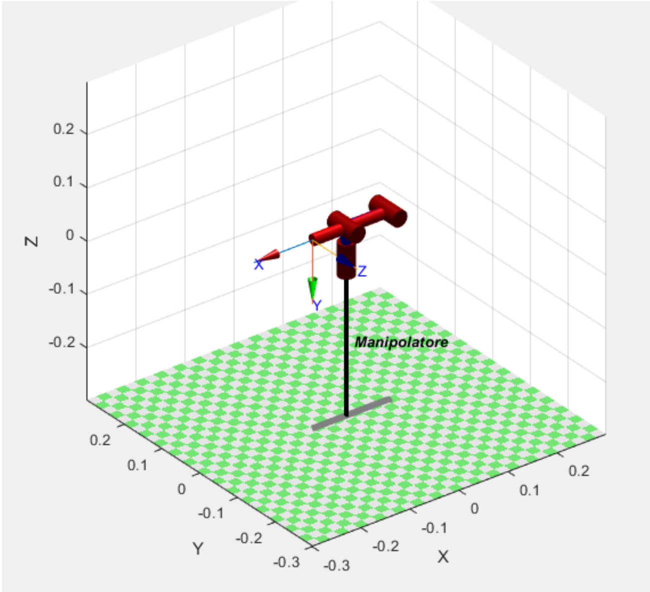

Figure 7: Elbow Singularity when $\theta_3 = \pi$

- $a_2 c_{\theta_2} + a_3 c_{\theta_2 + \theta_3} = 0$
  Using in particular:
  $\theta_2 = 24.351° = 0.425 rad$ and $\theta_3 = \cos^{-1}\left(\frac{-a_2}{a_3}\cos(\theta_2)\right) - \theta_2 = 95.839° = 1.627 rad$ we find the singularity. In this case, is possible to see that the origin of the end effector is passing through the axis of motion of the first joint.

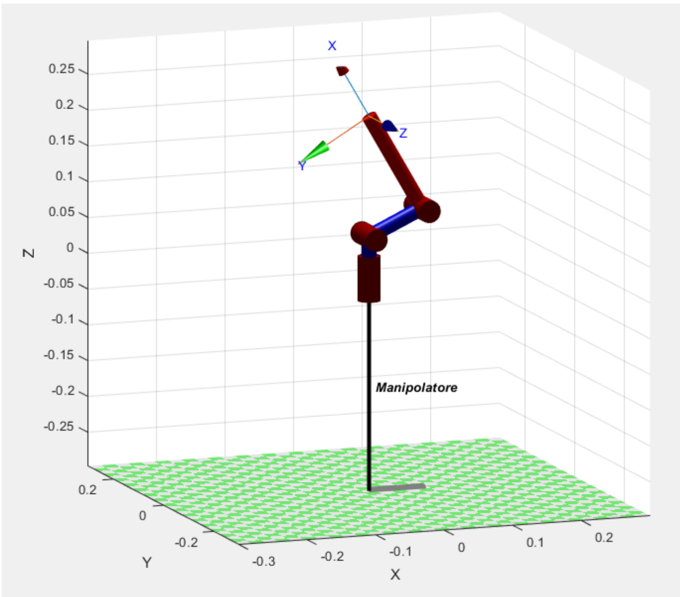

Figure 8: Shoulder Singularity

The values of the determinant when computing the singularities on the second link is not exactly zero but 1.1061e-19, which can still be considered as a singularity position.

Trajectory Planning

Before starting the study of the trajectory, we compare some known configurations between the Matlab representation of the robot and the real one and we find out that there is an offset in the third link of about 90°. In order to have a more accurate control of the robot, we also introduce a small offset of 36° in the second link. In this way, the MATLAB plot with some known angles is the same as the real position of the robotic arm. We have to find an interpolating function that put in relationship the values of the joint variables with a time sequence. First of all, we have to check if the robot is able to reach one desired final point starting from an initial one. After that, we can set the values of the joint variables in such a way to follow a desired trajectory.

First, we started by constructing a code that enables the robot to compute a trajectory that should allow to draw a letter D in the space. After conducting tons of tries, we were not able to draw a good D letter. The best result is the one on the picture above:

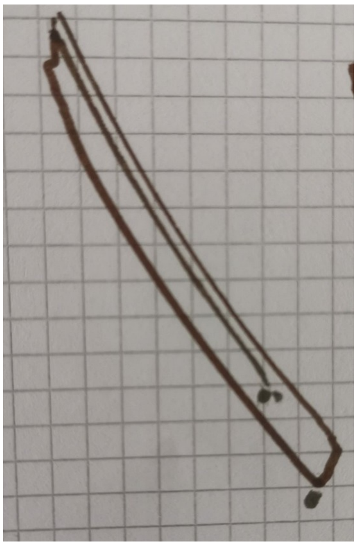

Figure 9: Robot drawing a D

The problem can be associated mainly to the scarce quality of the servomotors that do not allow a good accuracy and the friction between robot and the surface in which it tries to draw. For this reason, when the robot tries to compute for example a circular trajectory, does not succeed because it cannot realize small angles movements.

Nonetheless, after additional testing we discovered that actually the robot is perfectly capable of managing straight lines. Therefore, we decided to change task and instead of letter D we accomplish the trajectory of letter L that contains only lines.

To make the robot compute this type of trajectory we developed a code that is able to draw the vertical line first, and then another part was realized for the horizontal line. At this stage, we encountered several issues related to the difference in RF orientations between the MATLAB end effector and the robot end effector. For this reason, the only thing that we could do was to assign manually the joints variables on the real manipulator so that the end of the vertical line was perfectly matched with the start of the horizontal line. After that, we rotated only the base of the robot letting the values of the other joint angles equal to the values that they assumed in the last point of the vertical line. The first attempts were resulting in drawing a mirrored letter L, that means the horizontal line was drawn on the opposite side. This is due to the fact that the joint variable of the base servomotor is always almost equal to zero. This means that the base is able to run only in the left direction since the range of motion is between 0 and 180 degrees. In order to solve this problem, we decided to introduce a small offset of 36° in such a way that the robot is capable of turn rightward moving towards 0°. The results are shown on the following picture:

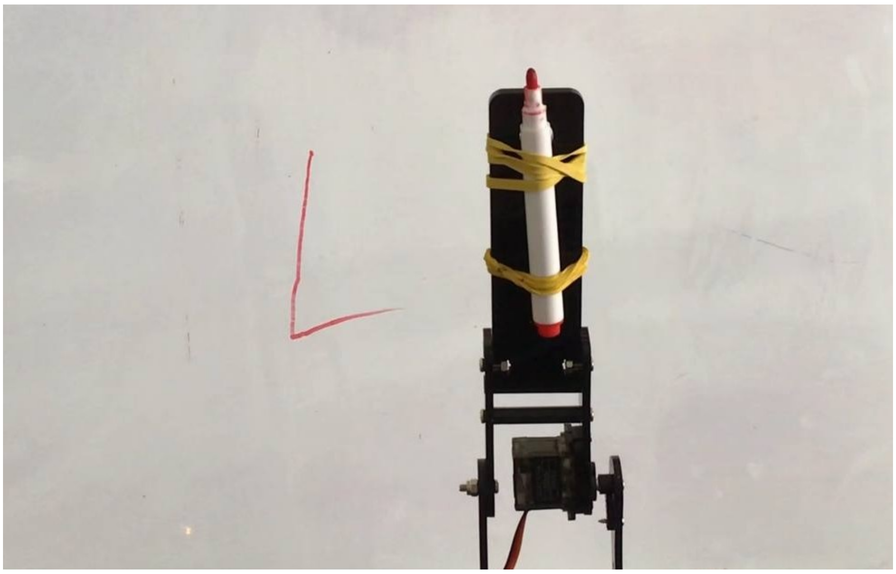

Figure 10: Robot successfully drawing a L

Conclusions

All in all, we can say that the task set for the project was accomplished but with lots of difficulties. At the beginning we thought that we could accomplish the writing of a letter such as D or R, but we realized that the robot we had was not capable of to do that. The reasons are to be attributed to the fact that the robot is not accurate enough to make small movements required to compute circular trajectory with good results. For what regards, straight lines instead the robot was quite good at doing them but problems such as loss in precision due to shacking, friction of the pen on the paper and imprecisions make the task to not be entirely accurate. Better results of course can be acquired with more expensive robot that have for example 6-axis of motion and so can guarantee a high level of dexterity.

Bibliography

Modelling, Planning and Control - L. Sciavicco, L. Villani, G. Oriolo, B. Siciliano

Robotics, Vision and Control, Fundamental Algorithms in MATLAB - Peter Corke