FSR HOMEWORK 2 2024

STUDENTE: PAOLO MAISTO
MATRICOLA: P38000191

Exercise 1:

a. Considering the following distribution: $\Delta_1 = \{[\,3x_1 \;\; 0 \;\; -1]^T\}, U \in R^3$

Immediately we can say that this distribution is **involutive** because any one-dimensional distribution is involutive and the zero vector belongs to any distribution by default.

Now, we aim to determine the annihilator for the distribution Δ_2, which involves finding a set of covectors w^*such that $w^*F(x) = 0$. It's worth noting that if $dim(\Delta_2) + dim(\Delta_2^{\perp}) = 3$ and $dim(\Delta_2) = 1$, then we can infer that $dim(\Delta_2^{\perp}) = 2$. So

$$\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \end{bmatrix}\begin{bmatrix} 3x_1 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \rightarrow \quad \begin{matrix} 3x_1w_1 - w_3 = 0 & \rightarrow & w_3 = 3x_1w_1 \\ 3x_1w_4 - w_6 = 0 & \rightarrow & w_6 = 3x_1w_4 \end{matrix}$$

Imposing for the independent variables: $w_1 = a_1, w_2 = a_2, w_4 = a_3, w_5 = a_4$. We achieve:

$$\Delta_1^{\perp} = \{[a_1 \quad a_2 \quad 3a_1x_1], [a_3 \quad a_4 \quad 3a_3x_1]\}$$

At this stage, we need to demonstrate that these two covectors are linearly independent. To achieve this, we construct the following matrix. Subsequently, we verify that <u>this</u> matrix is full rank (rank = 2) by considering a minor matrix of size 2 × 2. We must ensure that the determinant of this minor is non-zero:

$$\begin{bmatrix} a_1 & a_2 & 3a_1x_1 \\ a_3 & a_4 & 3a_3x_1 \end{bmatrix} \rightarrow \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \rightarrow \begin{vmatrix} a_1 & a_2 \\ a_3 & a_4 \end{vmatrix} = a_1a_4 - a_2a_3 \neq 0 \rightarrow a_1a_4 \neq a_2a_3$$

At the end, if we select parameters to satisfy the previous relation, then $\Delta_1^{\perp}$ is the annihilator for the distribution Δ_1.

b. Considering the following distribution: $\Delta_2 = \{[\,1\ \ 0\ x_2]^T, [\,0\ \ -1\ x_1]^T\}, U \in R^3$

First of all I construct the matrix F:

$$F = \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ x_2 & x_1 \end{bmatrix}$$

Considering the first 2×2 matrix, the determinant is -1, therefore the dimension of this distribution is 2 $\forall x$, thus the distribution is not singular.
The distribution Δ_2 has dimension $dim(\Delta_2) = 2$, now I proceed to compute the lie bracket:

$$\left[\begin{bmatrix} 1 \\ 0 \\ x_2 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \\ x_1 \end{bmatrix}\right] = [f_1, f_2] = \frac{\partial f_2}{\partial x} f_1 - \frac{\partial f_1}{\partial x} f_2$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ x_2 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} 0 \\ -1 \\ x_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

The next step is verify that lie bracket $[f_1, f_2]$ belongs to span genereted by vectors f_1, f_2. To accomplish this, we construct the following 3×3 matrix, where its columns are the vectors $f_1, f_2, [f_1, f_2]$:

$$F_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ x_3 & x_1 & 2 \end{bmatrix} \quad \rightarrow \quad det(F_1) = -2 \neq 0$$

So this matrix is always full rank: $rank(F_1) = 3$
Thus, we dedude that the vector $[f_1, f_2]$ is not a linear cobination of f_1 and f_2. So we can come to conclusion that distribution Δ_2 is **not involutive**.

Now, we aim to determine the annihilator for the distribution Δ_2, which involves finding a set of covectors w^*such that $w^*F(x) = 0$. It's worth noting that if $dim(\Delta_2) + dim(\Delta_2^\perp) = 3$ and $dim(\Delta_2) = 2$, then we can infer that $dim(\Delta_2^\perp) = 1$. So:

$$[w_1 \quad w_2 \quad w_3]\begin{bmatrix} 1 & 0 \\ 0 & -1 \\ x_2 & x_1 \end{bmatrix} = [0 \quad 0] \quad \rightarrow \quad \begin{aligned} w_1 + w_3 x_2 &= 0 \\ -w_2 + w_3 x_1 &= 0 \end{aligned} \quad \rightarrow \quad \begin{aligned} w_3 &= -\frac{w_1}{x_2} \\ w_2 &= w_3 x_1 = -\frac{w_1 x_1}{x_2} \end{aligned}$$

Imposing for the independent variables that $w_1 = a$. Thus, we obtain the following annihilator:

$$\Delta_2^\perp = \left\{\begin{bmatrix} a & -\frac{a x_1}{x_2} & -\frac{a}{x_2} \end{bmatrix}\right\}$$

It's worth noting that to avoid considering the banal solution $[0 \quad 0 \quad 0]$, it's necessary to impose the constraint $a \neq 0$.

c. Considering the following distribution: $\Delta_3 = \{[\,2x_3 \;\; -1 \; 0]^T, [x_2 \;\; x_1 \; 1]^T\}, U \in R^3$

First of all I construct the matrix F:

$$F = \begin{bmatrix} 2x_3 & x_2 \\ -1 & x_1 \\ 0 & 1 \end{bmatrix}$$

Considering the last 2×2 matrix, the determinant is -1, therefore the $rank(F) = 2$ and the dimension of this distribution is 2 $\forall x$, thus the distribution is not singular.
The distribution Δ_3 has dimension $dim(\Delta_3) = 2$, now I proceed to compute the lie bracket:

$$\left[\begin{bmatrix} 2x_3 \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} x_2 \\ x_1 \\ 1 \end{bmatrix}\right] = [f_1, f_2] = \frac{\partial f_2}{\partial x} f_1 - \frac{\partial f_1}{\partial x} f_2$$

$$= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2x_3 \\ -1 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_2 \\ x_1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 2x_3 \\ 0 \end{bmatrix} - \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -3 \\ 2x_3 \\ 0 \end{bmatrix}$$

The next step is verify that lie bracket $[f_1, f_2]$ belongs to span genereted by vectors f_1, f_2. To accomplish this, we construct the following 3×3 matrix, where its columns are the vectors $f_1, f_2, [f_1, f_2]$:

$$F_1 = \begin{bmatrix} 2x_3 & x_2 & -3 \\ -1 & x_1 & 2x_3 \\ 0 & 1 & 0 \end{bmatrix} \quad \rightarrow \quad det(F_1) = 3 - 4(x_3)^2 \neq 0$$

So, I can deduce that the matrix is always full rank by the fact that $rank(F_1) = \begin{cases} 3, & if \;\; x_3 = 0 \\ 3, & if \;\; x_3 \neq 0 \end{cases}$:

Thus, we dedude that the vector $[f_1, f_2]$ is not a linear cobination of f_1 and f_2. So we can come to conclusion that distribution Δ_3 is **not involutive**.

Now, we aim to determine the annihilator for the distribution Δ_3, which involves finding a set of covectors w^*such that $w^*F(x) = 0$. It's worth noting that if $dim(\Delta_3) + dim(\Delta_3^{\perp}) = 3$ and $dim(\Delta_3) = 2$, then we can infer that $dim(\Delta_3^{\perp}) = 1$. So:

$$[w_1 \quad w_2 \quad w_3] \begin{bmatrix} 2x_3 & x_2 \\ -1 & x_1 \\ 0 & 1 \end{bmatrix} = [0 \quad 0] \quad \rightarrow \quad \begin{matrix} 2w_1x_3 - w_2 = 0 \\ w_1x_2 + w_2x_1 + w_3 = 0 \end{matrix}$$

$$\begin{matrix} w_2 = 2w_1x_3 \\ w_3 = -w_1x_2 - w_2x_1 \end{matrix} \quad \rightarrow \quad \begin{matrix} w_2 = 2w_1x_3 \\ w_3 = -w_1x_2 - 2w_1x_3x_1 \end{matrix}$$

Imposing for the independent variables that $w_1 = a$. Thus, we obtain the following annihilator:

$$\Delta_3^{\perp} = \{[a \quad 2ax_3 \quad -ax_2 - 2ax_3x_1]\}$$

It's worth noting that to avoid considering the banal solution $[0 \quad 0 \quad 0]$, it's necessary to impose the constraint $a \neq 0$.

<u>Exercise 2:</u>

The omnidirectional mobile robot we're looking at has these generalized coordinates:

$$q = [x \;\; y \;\; \theta \;\; a \;\; \beta \;\; \gamma]^T \quad \rightarrow \quad n = 6$$

Where:

- x, y are centre coordinate of robot;
- θ is orientation of robot;
- a, β, γ are angular position each wheel;

And I define:

- $r = 0.4$ is radius of the wheels;
- $l = 0.6$ is distance between wheels and centre of robot.

To establish the kinematic model for this robot, we must determine the matrix G such that $G(q) \in \mathcal{N}(A^T(q))$. To achieve this, we employ the MATLAB command null: $G = null(A)$.
A^T is a matrix with three rows and six columns, therefore G is a matrix with six rows and three columns. The kinematic model is:

$$\dot{q} = G(q)u = G(q)\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

Defining each column of G like g_1, g_2, g_3. The accessibility distribution Δ_a is constructed with these three vectors, where each lie bracket is comprised of g_1, g_2, and g_3.
The Jacobian of these vectors needs to be calculated first for this reason:

```
% Computing the Jacobian refering the 6 coordinates
dg1=jacobian(g1,q);
dg2=jacobian(g2,q);
dg3=jacobian(g3,q);
```

And then, we can compute the lie bracket as:

```
% Computing the lie bracket
Lg12=dg2*g1-dg1*g2;
Lg13=dg3*g1-dg1*g3;
Lg23=dg3*g2-dg2*g3;
```

Notice that if we construct a matrix whose columns are vectors of g_1, g_2, g_3, L_{12}, L_{13}, and L_{23}, its rank is five. Therefore, we can infer that one vector is linearly dependent. By removing this vector, the accessibility distribution has a dimension $v = \dim(\Delta_a) = 5$. Consequently, we can say:

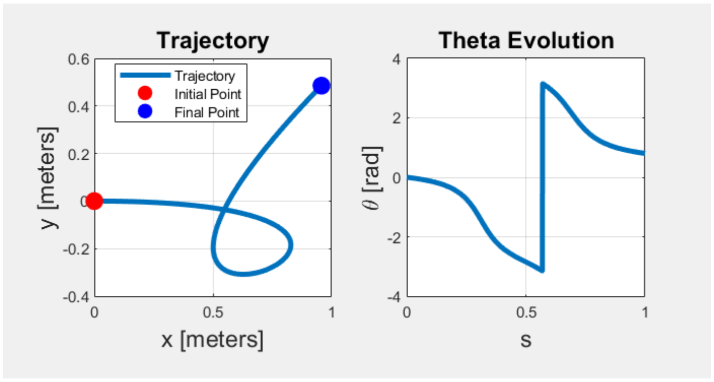
$$\begin{matrix} m = 3 \\ v = 5 \\ n = 6 \end{matrix} \quad \rightarrow \quad m < v < n$$

The system is still **nonholonomic** and $A^T(q)\dot{q} = 0$ have only partially integrable constraints (as it is printed on the screen by the code in matlab, the script is called <u>*‘Es_2’*</u>).

Exercise 3:

We employ a cubic Cartesian polynomial to execute a path planning algorithm, aiming to guide the unicycle from configuration $q_0 = [0\ \ 0\ \ 0]^T$ to the random final point (at each run the final point coordinates are printed on the screen as shown in the following image). The file called 'Es_3_1' implements only the cubic Cartesian polynomial, the results are shown in the following graphics. It is evident that the unicycle goes from the origin to the final point which has the following coordinates (0.95717,0.48539) with an orientation of $0.800208\ rad \cong\ 45{,}85°$.

```
Coordinate of final point: (0.95717, 0.48538, 0.80028)
```



However, if we encounter velocity constraints as follows, it becomes necessary to implement a specific time law to adhere to these constraints.

- $|v(t)| \leq 2\ m/s$
- $|\omega(t)| \leq 1 rad/s$

The 'Es_3_2' file implements the second part of this exercise. In which I define a third order polynomial time law with the following conditions:

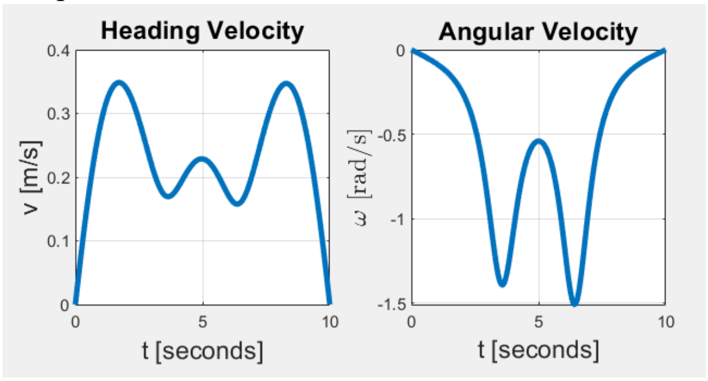
- Start instant $t_i = 0$ and final instant is made to choose through terminal, to every iteration if the t_f is too small for the constraints to be respected, it returns error with the graphs that emphasize that the constraints are exceeded, for then to ask to insert a greater t_f always through terminal;
- Start position $s(t_i) = s(0) = 0$ and random final position;
- Start velocity $\dot{s}(t_i) = \dot{s}(0) = 0$ and final velocity $\dot{s}(t_f) = 0$;

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \qquad with\!:\ T = t_f - t_i$$

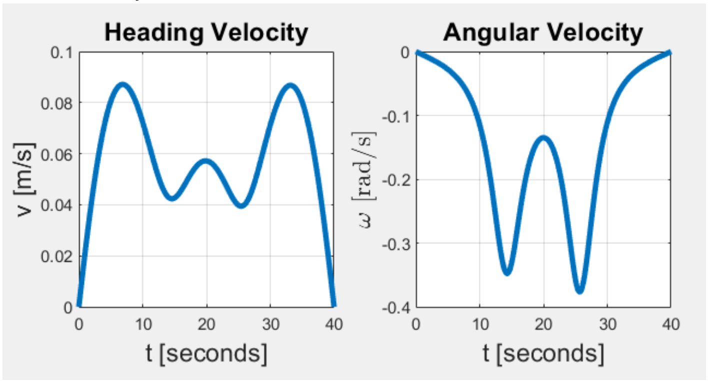
Where:

$$\begin{cases} a_0 = s(t_i) \\ a_1 = \dot{s}(t_i) \\ a_2 = \dfrac{-3\left(s(t_i) - s(t_f)\right) - \left(2\dot{s}(t_i) + \dot{s}(t_f)\right)T}{T^2} \\ a_3 = \dfrac{2\left(s(t_i) - s(t_f)\right) + \left(\dot{s}(t_i) + \dot{s}(t_f)\right)T}{T^3} \end{cases} \rightarrow \begin{cases} a_0 = 0 \\ a_1 = 0 \\ a_2 = \dfrac{3}{t_f^2} \\ a_3 = -\dfrac{2}{t_f^3} \end{cases}$$

As shown in the following pictures initially I put as input a time interval of $[t_i, t_f] = [0{,}10]\ s$, but velocity bounds are not respected.



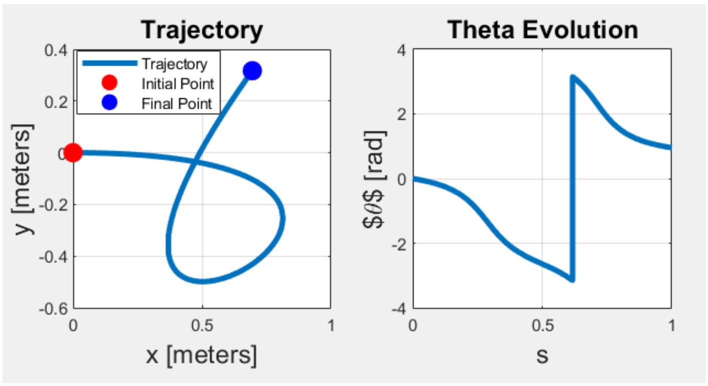
So I inserted a higher final time $t_f = 40\ s$ and consequently velocity constraints are respected.



The following terminal first shows the coordinates of the final point and then shows the requests for the final time. Obviously the end point of the two cases ('Es_3_1' and 'Es_3_2') differs because they are different simulations and in both simulations the end point is generated randomly.

```
Coordinate of final point: (0.69483, 0.3171, 0.95022)
Insert tf value: 10
It is recommended to increase the T period
Insert tf value: 40
All constraints have been satisfied
```

In the same way as in the previous case, the trajectory and the evolution of theta are reported as shown in the following figure.

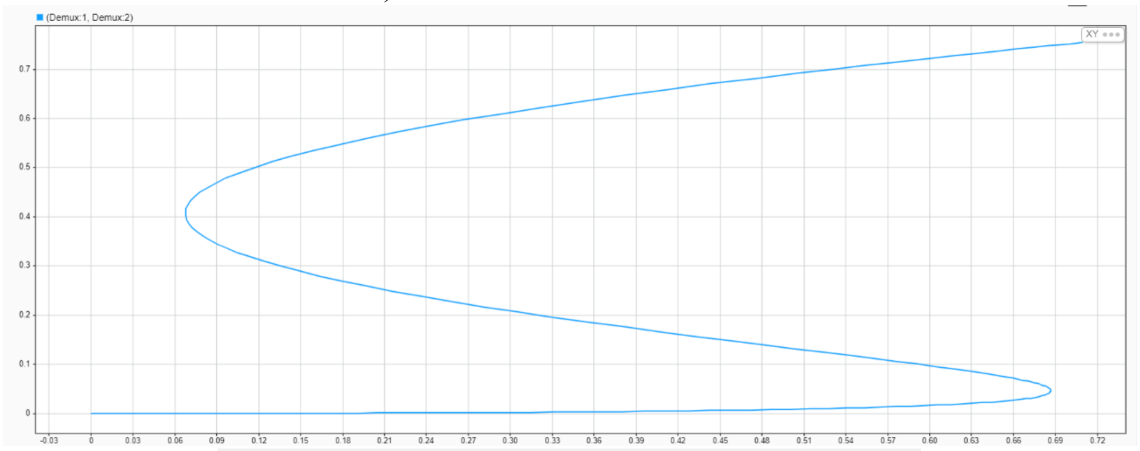


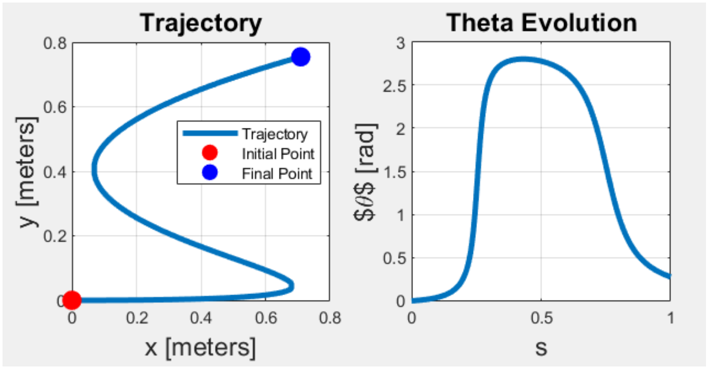
Exercise 4:

In order to implement input/output linearization control we utilize the trajectory from the previous exercise as a reference. This approach allows us to control the position of the unicycle. The method relies on feedback from the following outputs, representing the Cartesian coordinates of a point B situated along the sagittal axis of the unicycle at a distance $|b|$. I have opted to set $b = 0.5$ to position it forward.

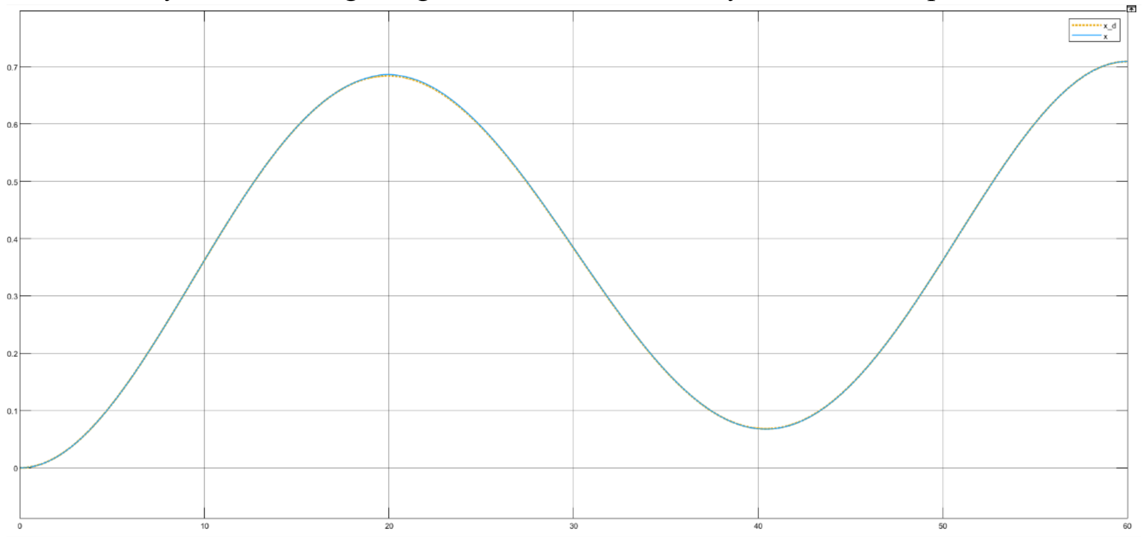
$$y_1 = x + bcos(\theta)$$
$$y_2 = y + bsin(\theta)$$

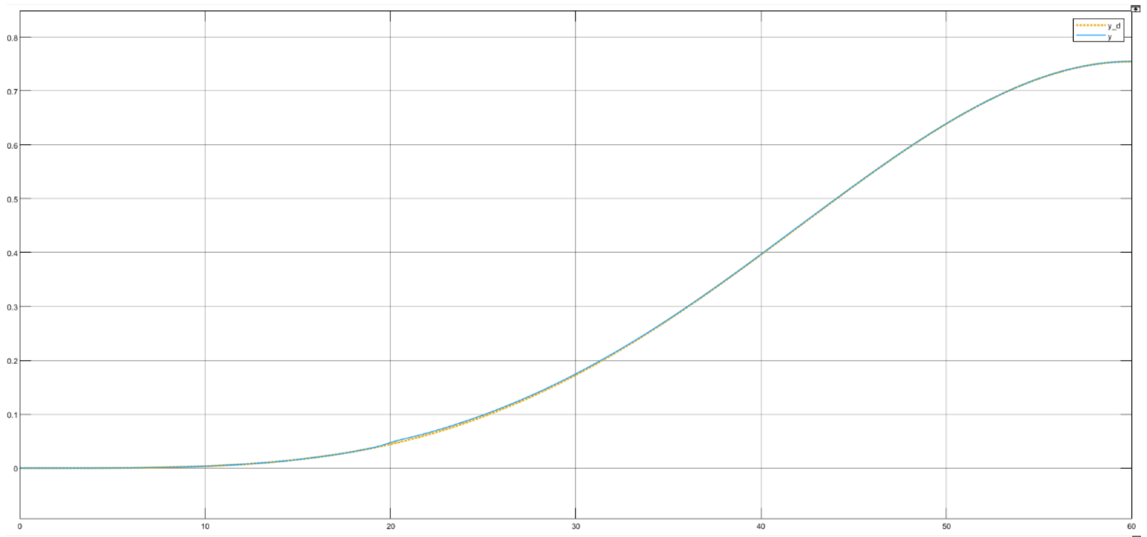
In the Simulink file'Es_4', I've implemented two blocks to compute these outputs. The first block calculates the reference signals y_{1d} and y_{2d}, which correspond to the trajectory generated in the previous exercise. The second block computes y_1 and y_2 for the unicycle. Subsequently, the controller calculates u_1 and u_2 using predefined formulas, where I defined the gain values $k_1 = k_2 = 5$. Finally, the last block computes the heading and angular velocity.
First of all, we observe the simulink results in the first image, while in the second are reported the results obtained by simulating the matlab file from which then the trajectory is resumed as a reference, so we can say that the trajectory is executed correctly (also because it was done in such a way that when the simulink file starts running, it automatically runs first the matlab file and then continue only after it has taken the data interessed).

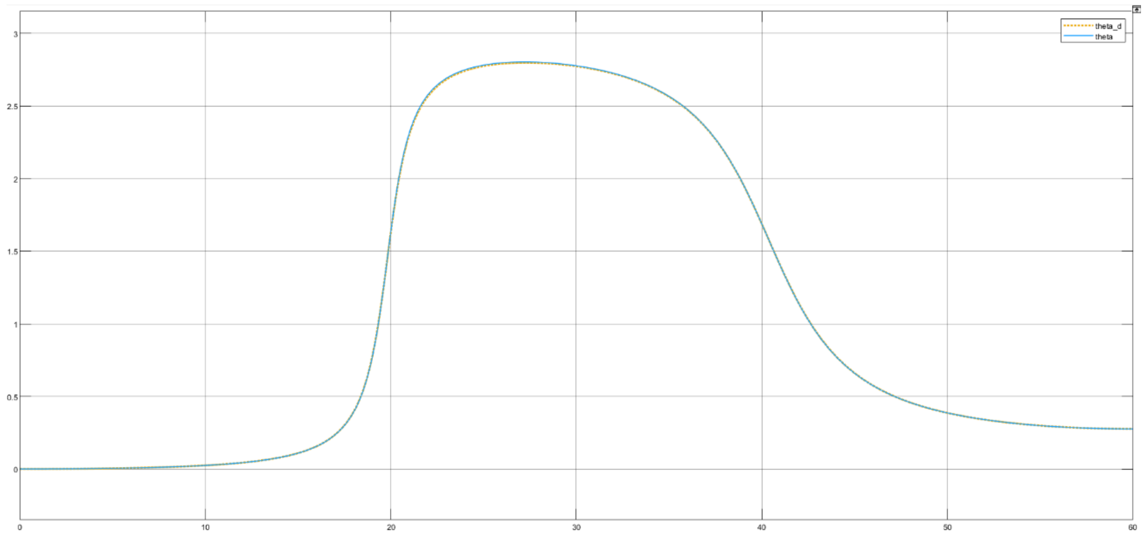




In fact as shown by the following images all variables correctly follow the respective references:

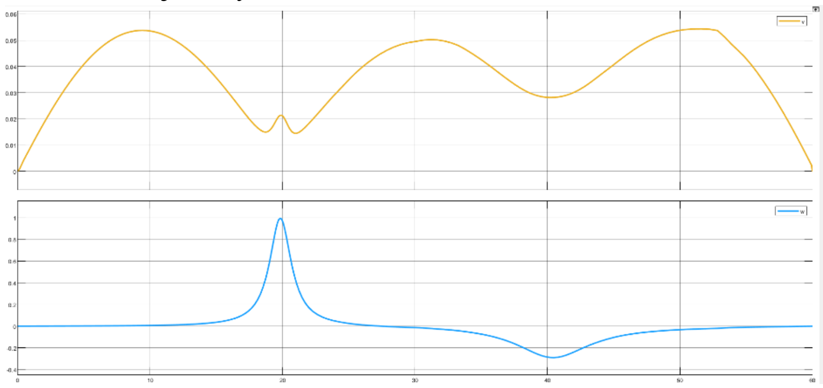


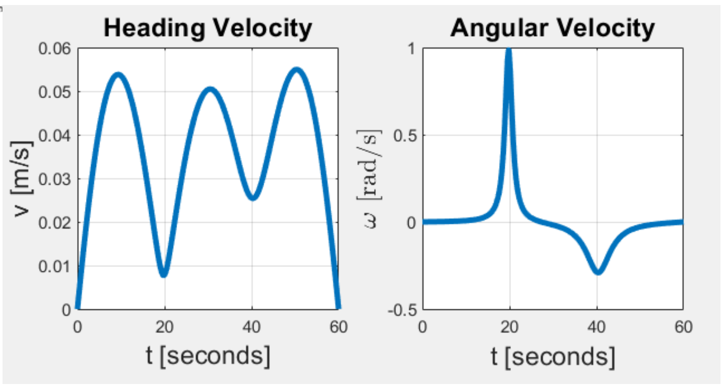




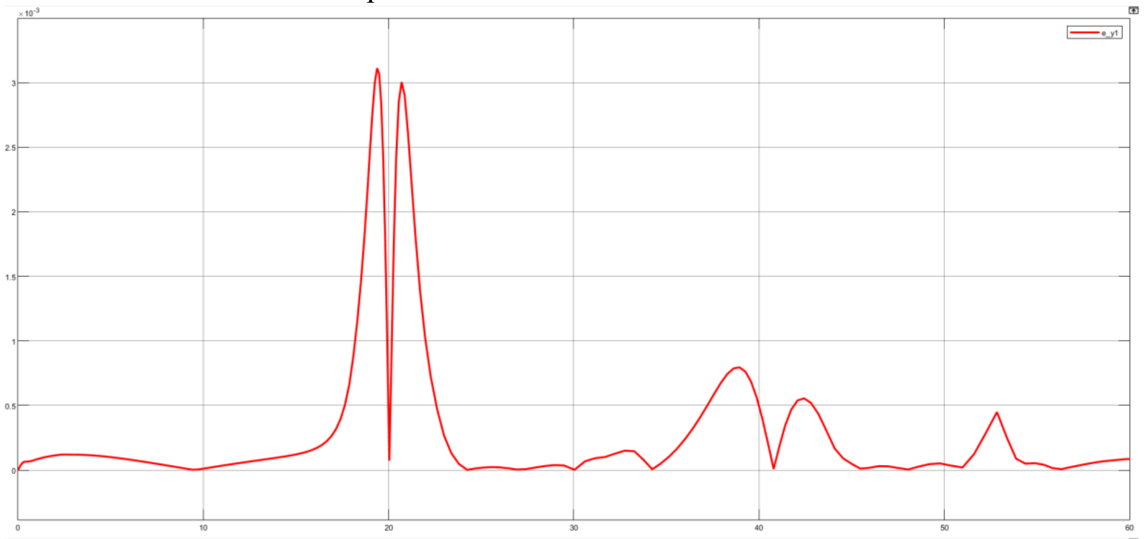
Obviously being each simulation on its own, because each of them is generated a different end point; the final point of exercise 4 will be once again different from the exercise 3_2 which will be once again different from the exercise 3_1.

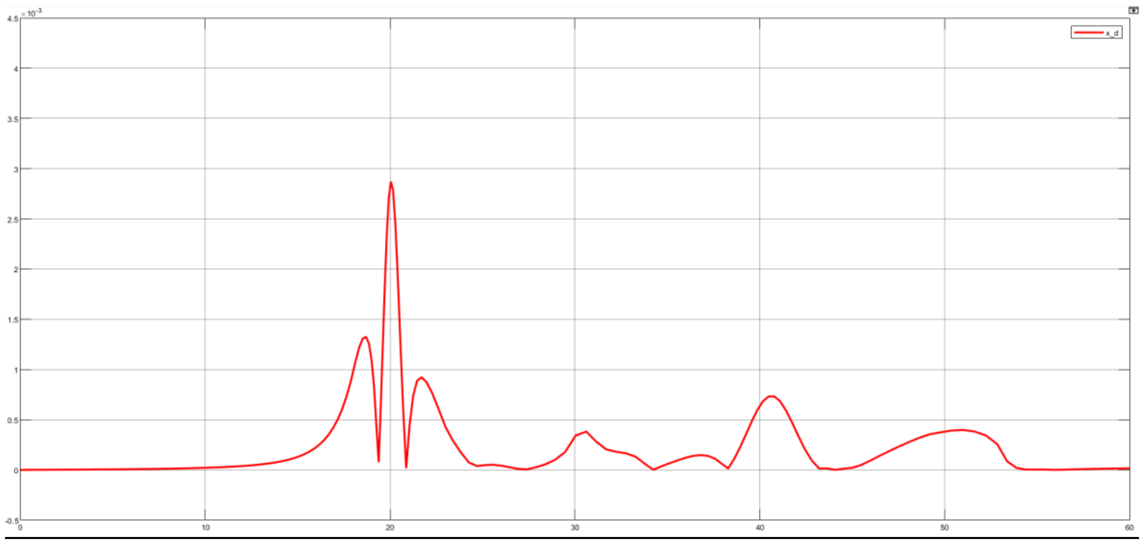
Before proceeding with the errors, the graphs concerning heading and angular velocity are also reported, to underline that the constraints imposed in the previous exercise are respected. As before, on the left are the results obtained by simulink, while on the right are the matlab file from which the trajectory reference is taken:

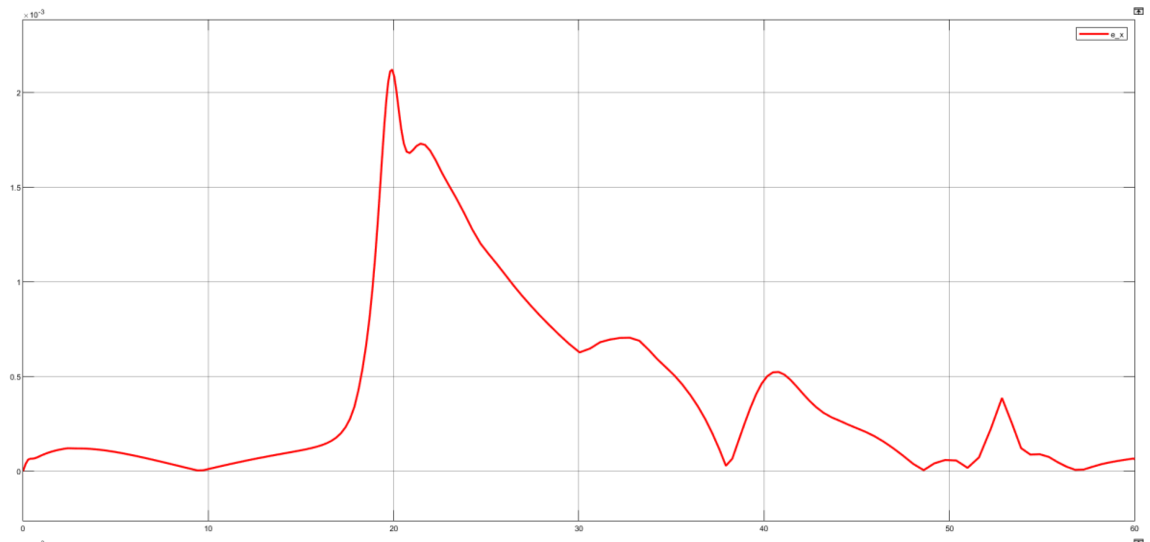


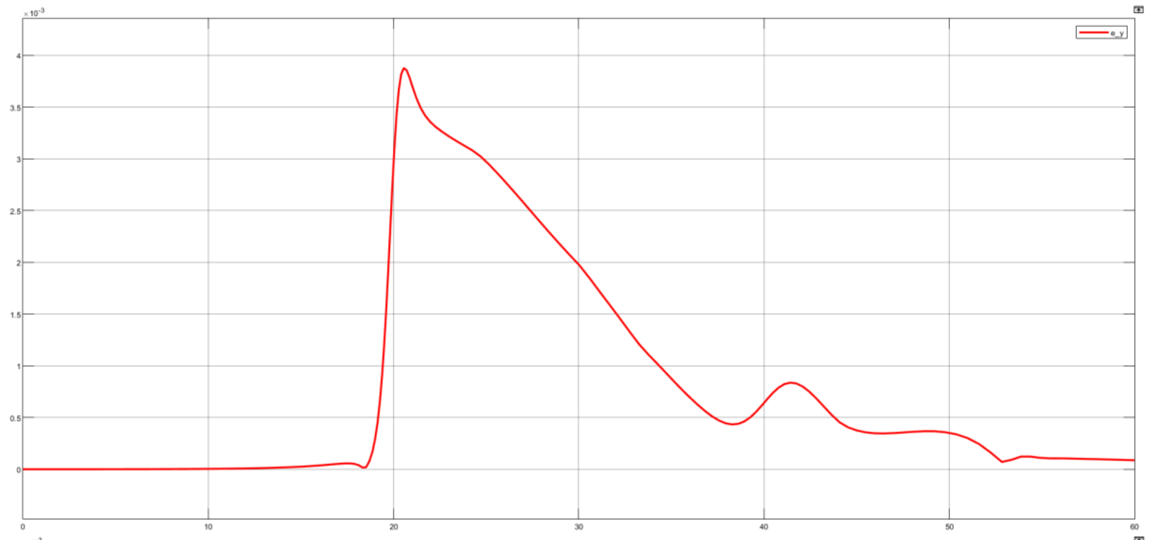


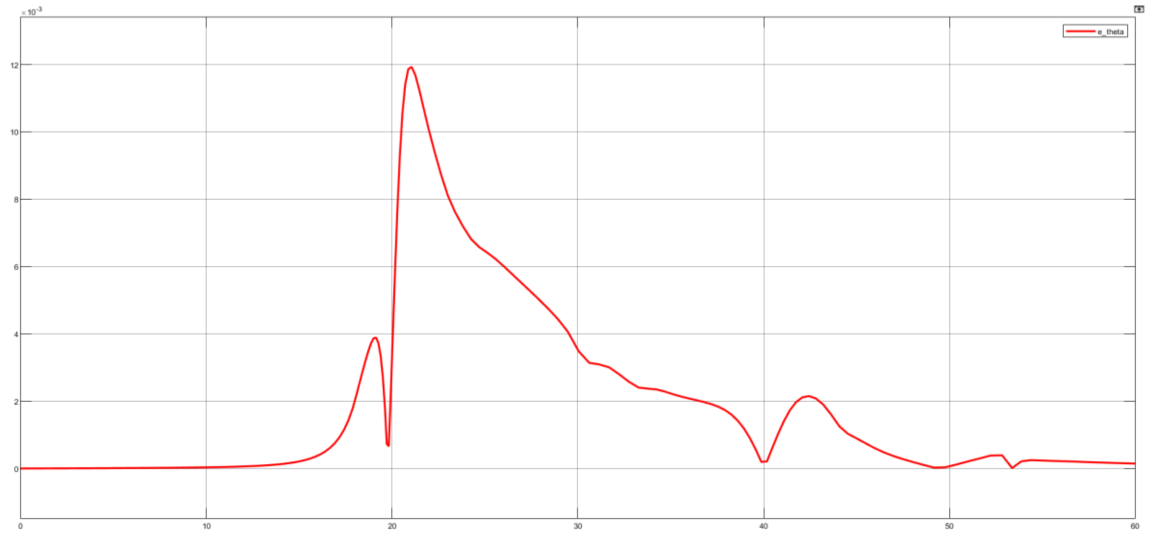
At the end the errors $e_{y1}, e_{y2}, e_x, e_y, e_\theta$, shown in the following pictures, are of the order 10^{-3}, therefore we can define them acceptable.











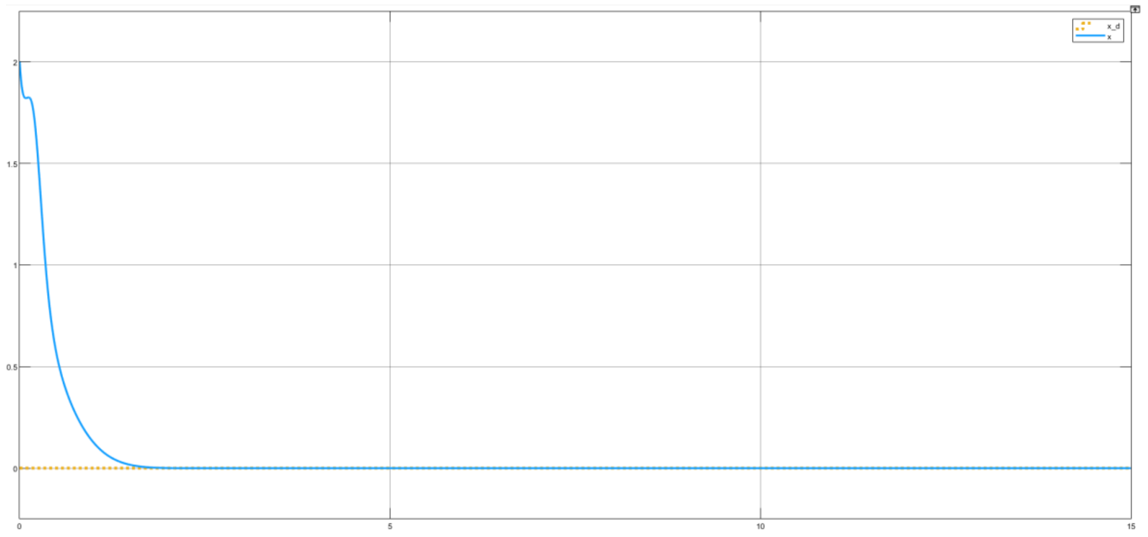
Analyzing the whole, in theory, this technique controls only the position but not the orientation of the unicycle. However, in practice, to accurately follow the desired trajectory, the unicycle must orient itself in a manner similar to θ_d. Despite this observation, if we examine the error on θ, we find that it consistently remains on the order of 10^{-3}, similar to the one for variables x and y, but slightly larger. Specifically, the maximum errors on x and y are $2.5 \cdot 10^{-3}$and $4 \cdot 10^{-3}$respectively, while the maximum error in θ is $12 \cdot 10^{-3}$.

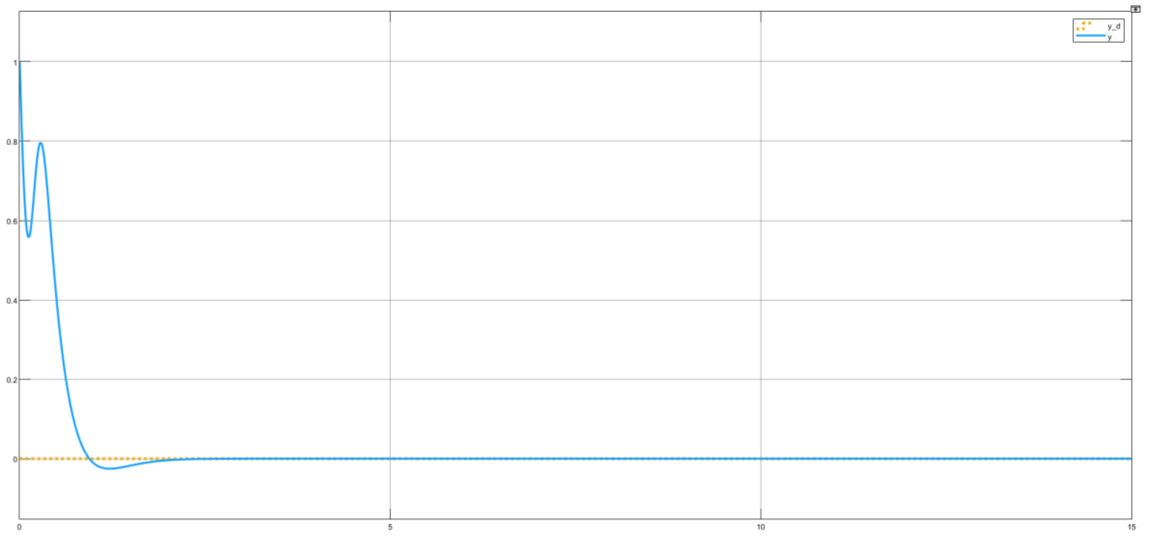
Exercise 5:

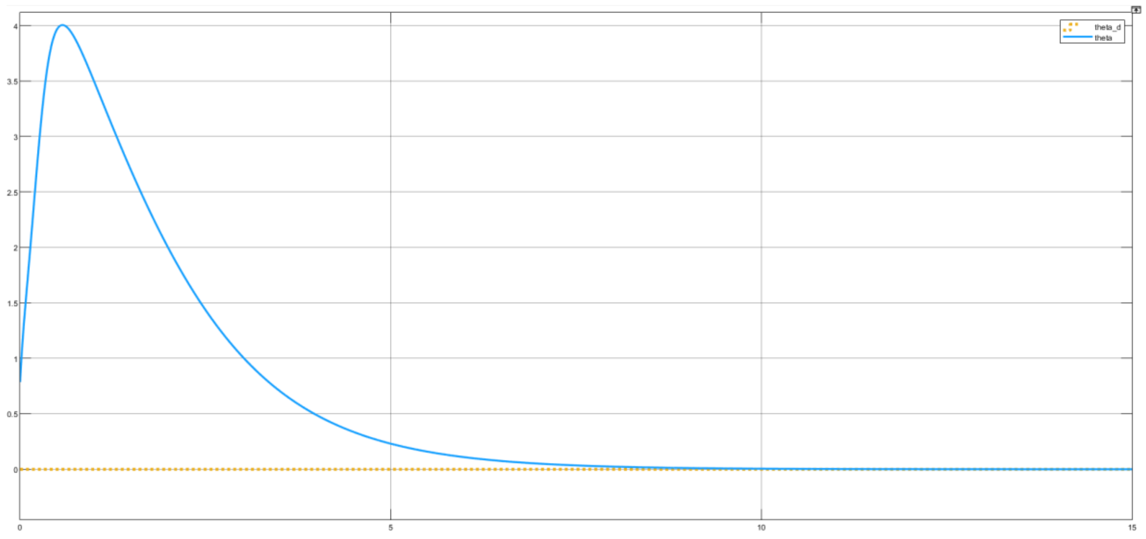
Our goal is to maneuver our unicycle robot from the $q_i = [2 \;\; 1 \;\; \pi/4]^T$ to $q_f = [0 \;\; 0 \;\; 0]^T$ in order to implement the unicycle posture regulator based on polar coordinates (using Simulink software). The initial step involves initializing the Unicycle Kinematic Model block with q_i. Subsequently, I have implemented a MATLAB function to execute the 2nd Order Runge-Kutta Approximation, with a chosen simple time of $T_s = 0.001\, s$. In order to design the following feedback controller:

$$\begin{cases} v = k_1 \rho cos(\gamma) \\ \omega = k_2\gamma + k_1 sin(\gamma)cos(\gamma)\left(1 + k_3\dfrac{\delta}{\gamma}\right) \end{cases} \quad with: \;\; k_1, k_2, k_3 > 0$$

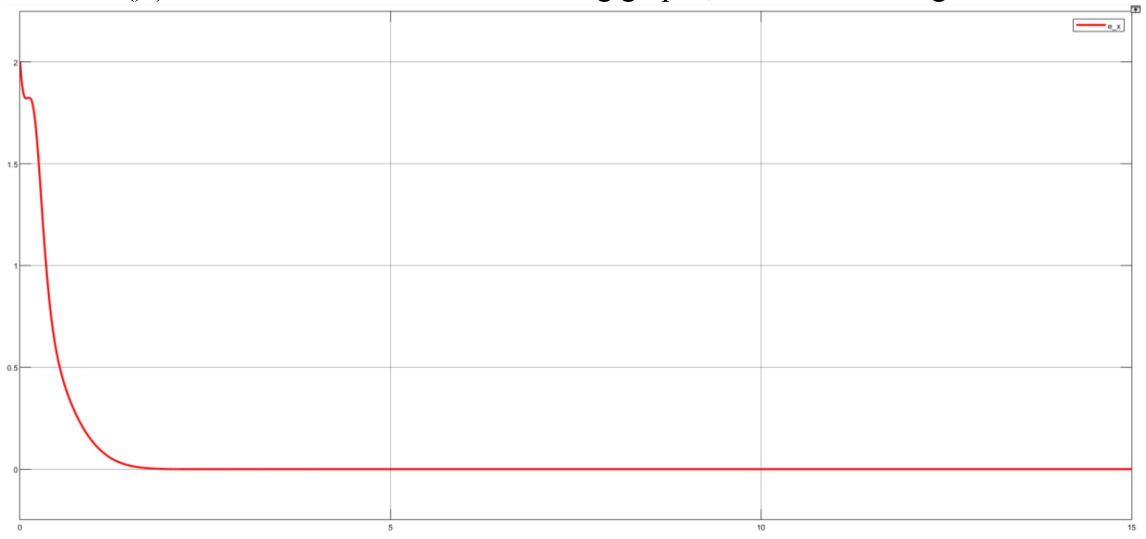
I have also deployed a MATLAB function to implement Cartesian/Polar Conversion. Subsequently, by defining the gains $k_1 = 4, k_2 = 5, k_3 = 0.2$, we can observe the following plots for x, y, θ. As evident from the plots, all variables reach 0 in a simulation of 15 seconds, indicating the attainment of the desired posture.

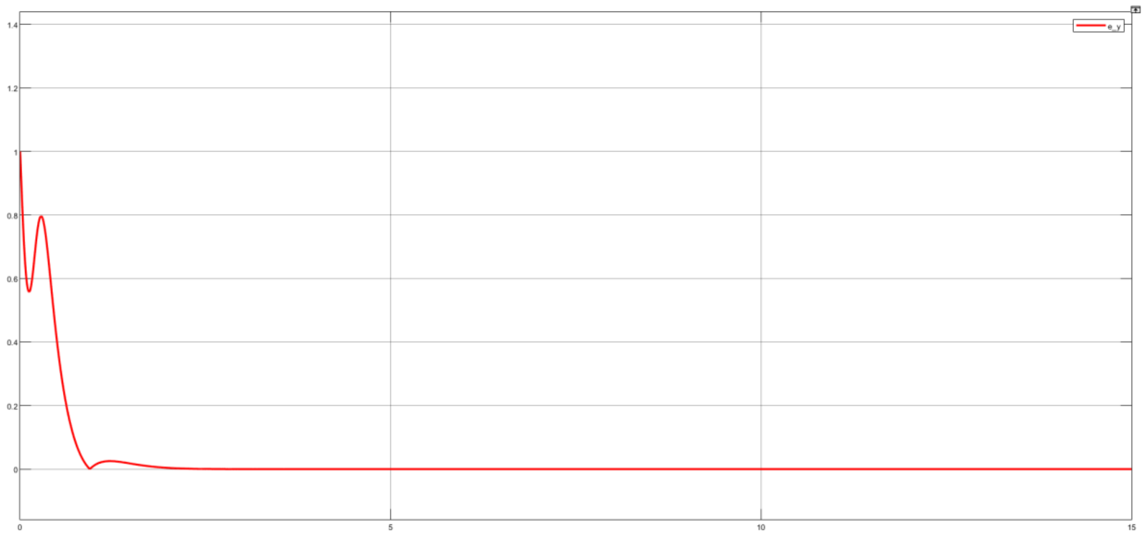


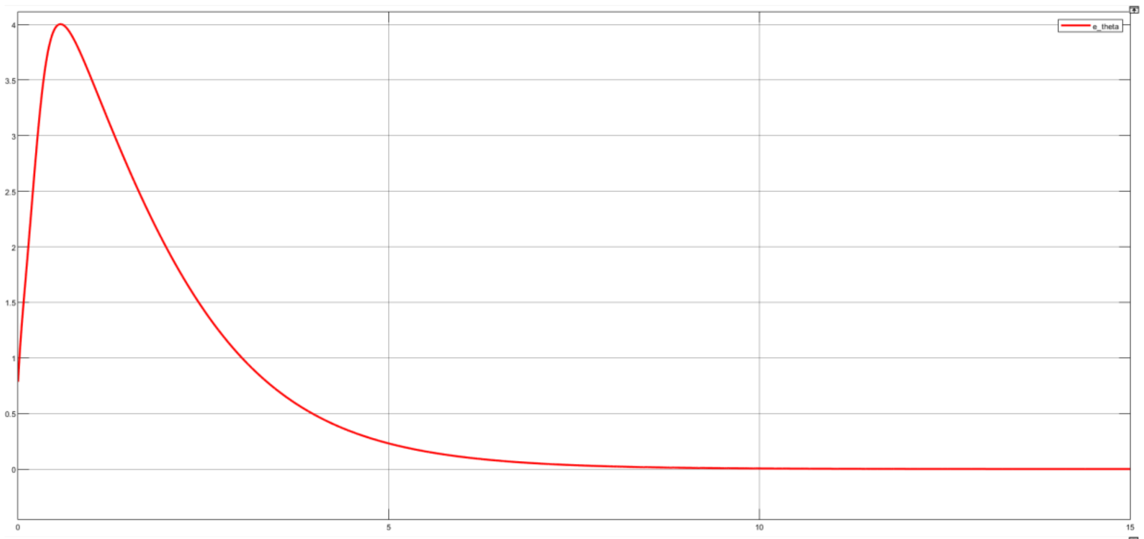


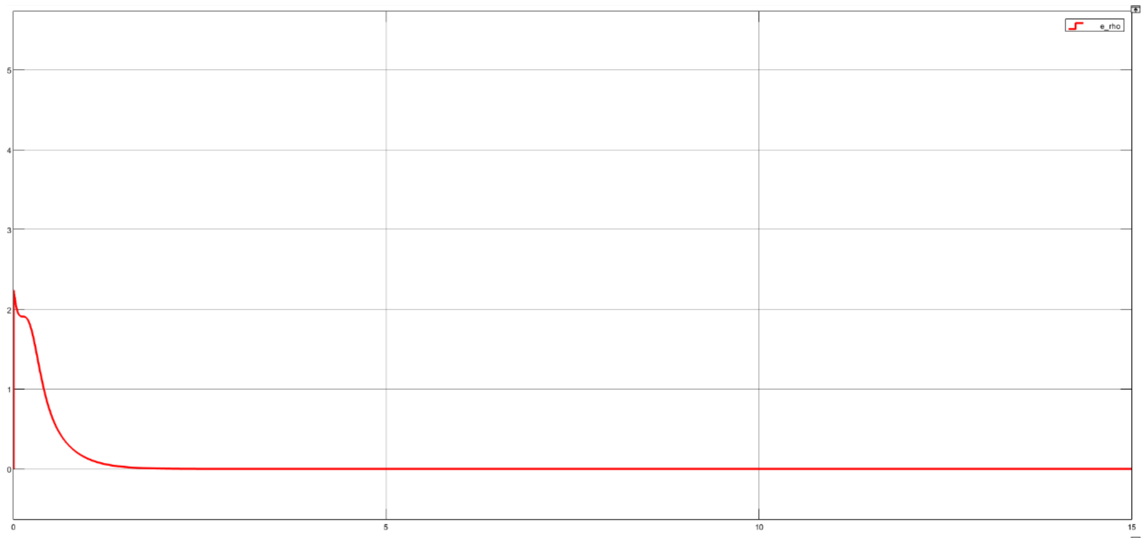


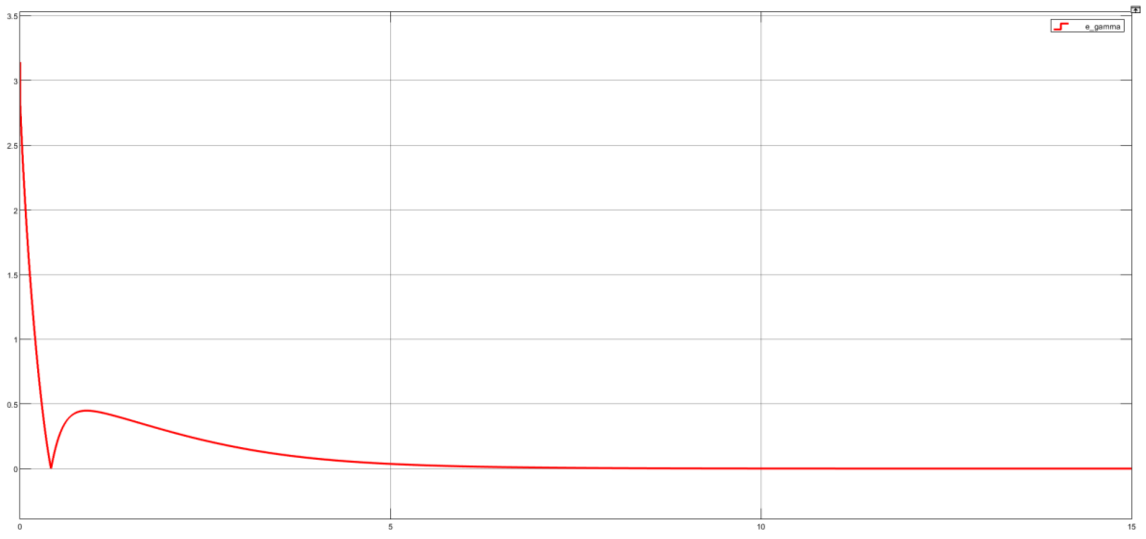
To confirm the success of the adjustments, we assess the errors on $x, y, \theta, \rho, \gamma, \delta$. At the end, for accurate error evaluation, the norm of the error was computed for each variable (using the MATLAB function ‘norm()’). As we can see from the following graphs, all errors converge to zero.

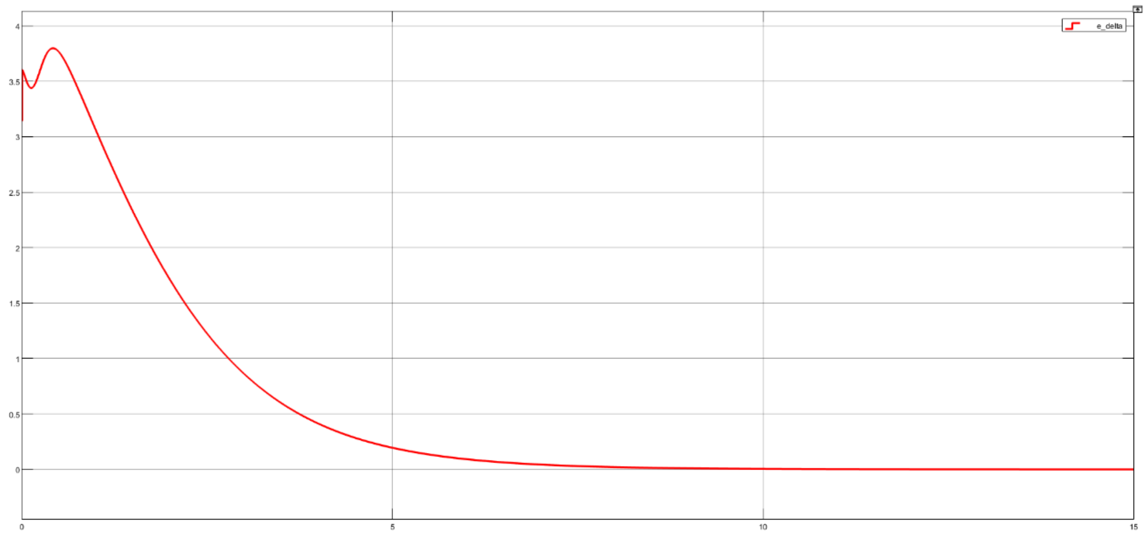





e_theta
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15


e_rho
5
4
3
2
1
0
0
5
10
15


e_gamma
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15


e_delta
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15