Implementation of a hierarchical controller and APF-algorithm to a quadrotor UAV

Antonio D'Angelo P38000115

July 21, 2024

1 Introduction

In this project we'll see an implementation of a hierarchical control and the artificial potential field algorithm to a dynamic model of an UAV quadrotor. Such dynamic model of the quadrotor is given by Professor Fabio Ruggiero which is implemented completely in Simulink and it is controlled by a geometrical controller.
This work is divided in two: the first half is the analysis and implementation of the hierarchical controller (instead of the geometrical). It will show how this is implemented and the simulation obtained, making also a comparison of the geometrical control (the planner in this half will be left untouched).
The second half is the implementation of the APF-algorithm and the creation of a virtual map with some cylindrical obstacle. In this half will be shown the forces calcuted with this new algorithm and also the trajectory taken by the UAV in the presence of an obstacle in its path (always with a comparison between the hierarchical and the geometrical).
One last note is to be made for the video registered for the work. They can be found at the link in the footnote [1]

2 Hierarchical control

One of the many way to control a UAV, in particular a quadrotor, is to use a hierarchical controller. It is designed to create two linear subsystems (one for the linear part and the other for the angular part) coupled by a non-linear interconnection term; basically the closed loop system it is composed by an inner loop witch controls the angular part, and an outer loop which controls the linear part interconnected with the inner loop by the aforementioned non-linear term (note that the inner loop goes through a feedback linearization).
The starting point is to express the model of the aerial robot in the RPY coordinates, having the following system:

$$\begin{cases} m\ddot{p}_b = mge_3 - u_T R_b e_3 \\ M(\eta_b)\ddot{\eta}_b = -C(\eta_b, \dot{\eta}_b)\dot{\eta}_b + Q^T(\eta_b)\tau^b \end{cases}$$

where:

- u_T and τ_b are respectively the control force and the control torques
- $e_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$
- m, g, I_b are respectively the mass, the gravity acceleration and the diagonal inertia matrix
- $M(\eta_b) = Q^T(\eta_b) I_b Q(\eta_b)$ symmetric and positive
- $C(\eta_b, \dot{\eta}_b) = Q^T(\eta_b) S(Q(\eta_b)\dot{\eta}_b) I_b Q(\eta_b) + Q^T(\eta_b) I_b \dot{Q}(\eta_b)$ with S the skew-symmetric operator

[1] Github video

- $Q(\eta_b) = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix}$
- η_b is the vector of the three Euler angles ϕ, θ, ψ

If the tracking error for the attitude is taken into consideration and also its derivative and the tracking error of the position and the linear velocity, it is easy to find out the following expression:

$$\begin{cases} \begin{bmatrix} \dot{e}_p \\ \ddot{e}_p \end{bmatrix} = A \begin{bmatrix} e_p \\ \dot{e}_p \end{bmatrix} + B(\mu_d - \ddot{p}_{b,d}) + \frac{1}{m} u_T \Delta \\ \\ \begin{bmatrix} \dot{e}_\eta \\ \ddot{e}_\eta \end{bmatrix} = A \begin{bmatrix} e_\eta \\ \dot{e}_\eta \end{bmatrix} + B(\tilde{\tau} - \ddot{\eta}_{b,d}) \end{cases}$$

with $A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{6X6}, B = \begin{bmatrix} 0 \\ I \end{bmatrix} \in \mathbb{R}^{6X3}$ and Δ the interconnection term between the linear and the angular part.
Lastly knowing that

$$u_t = m\sqrt{\mu_{d,x}^2 + \mu_{d,y}^2 + (\mu_{d,z} - g)^2}$$
$$\tau^b = I_b Q(\eta_b)\tilde{\tau} + Q(\eta_b)^{-T} C(\eta_b, \dot{\eta}_b)\dot{\eta}_b$$

and choosing μ_d and $\tilde{\tau}$ as follow:

$$\begin{cases} \mu_d = -K_p \begin{bmatrix} e_p \\ \dot{e}_p \end{bmatrix} + \ddot{p}_{b,d} \\ \\ \tilde{\tau} = -K_e \begin{bmatrix} e_\eta \\ \dot{e}_\eta \end{bmatrix} + \ddot{\eta}_{b,d} \end{cases}$$

it is insured that the errors goes to asymptotic stability equal to zero (naturally K_p and K_e are positive definite).
A side note is to be made regarding the complication using this type of controller; the primary one being the occurrence of singularities. These singularities are the pitch angle θ reaching $\pm\frac{\pi}{2}$ and the thrust $|u_z| < g$. In subsequent experiments they will not be shown, in other words, it will be completely avoided those cases. Another problem it arises was the use of various derivates, thus introducing noises into the system; this case also will not be shown due to the smoothness of the signals given by the planner.

As said before to achieve a hierarchical control the model must be expressed in RPY coordinates, but the UAV dynamic model retrieved by the Professor is written in coordinate-free, thus using rotation matrix R_b and angular velocity ω_b^b. While retrieving the position, the linear velocity, the acceleration, the angular velocity, and the angular acceleration was simple (they were already given by the model itself), the Euler angles were obtained without the use of any derivation, but simply employing the definition of the Q matrix. In fact the Q matrix is the transformation matrix such that $\omega_b^b = Q(\eta_b)\dot{\eta}_b$ and a simple inversion it was easy to obtain the angles[1].

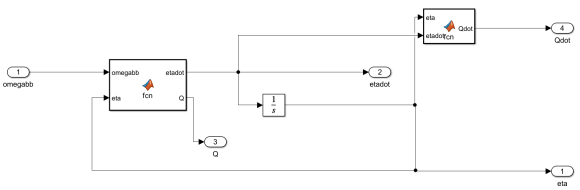
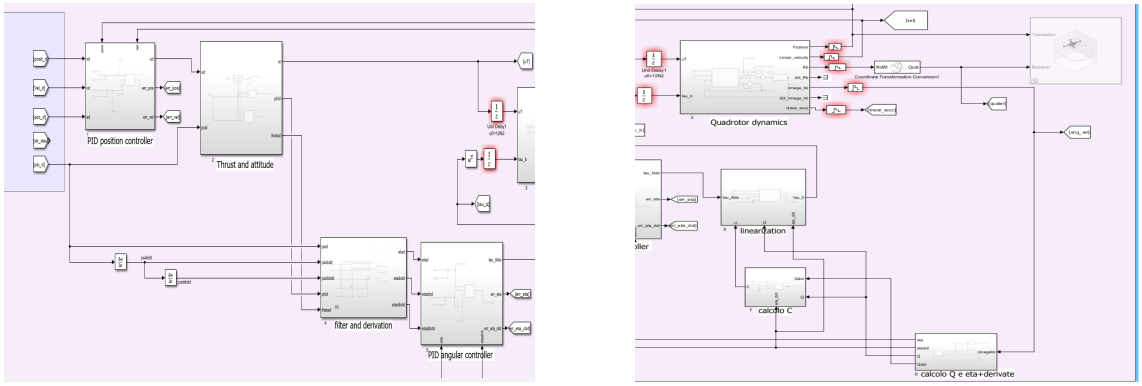


Figure 1: Simulink scheme for retrieving Euler angles

Lastly it is shown a completely Simulink of the entire controller[2].



(a) (b)

Figure 2: Hierarchical control of a quadrotor UAV in Simulink

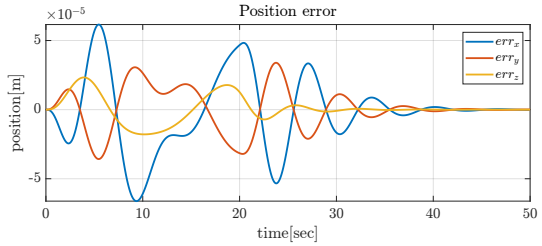
3 First results

The first experiments are made using the following set of parameters:

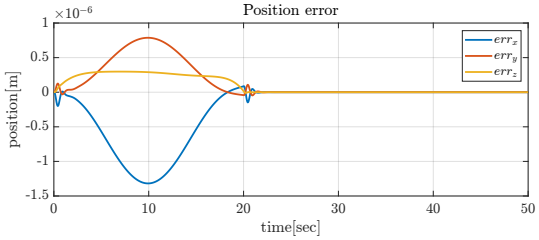
- $x_0 = 1 y_0 = -1 z_0 = -5$ and $x_f = 6 y_f = -4 z = -7$ as the staring position and final position[2]. The starting position was not chosen $\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$ but a position in which the UAV is already in the air, because later using the APF-algorithm the ground is chosen as an obstacle, thus having some complication starting in the origin.
- $\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$ the starting and final linear velocity, as well the linear acceleration and initial attitude
- $\frac{pi}{9} rad$ as the final angle ψ
- $T_s = 0.001$ as the sampling time
- $T_s tart = 0 T_f inal = 20$ and $T_e nd = 50$ respectively the starting instant, the instant to reach the final point, and the instant to evaluate the steady-state.
- $newKP = \begin{bmatrix} K_p & K_v \end{bmatrix}$ with $K_p = diag(\begin{bmatrix} 1 & 1 & 1 \end{bmatrix})$ and $K_v = diag(\begin{bmatrix} 0.5 & 0.5 & 0.5 \end{bmatrix})$ as the gains for the outer-loop controller
- $newKE = \begin{bmatrix} K_{etap} & K_{etav} \end{bmatrix}$ with $K_{etap} = diag(\begin{bmatrix} 1 & 1 & 1 \end{bmatrix})$ and $K_{etav} = diag(\begin{bmatrix} 0.7 & 0.7 & 0.7 \end{bmatrix})$ as the gains for the inner-loop controller

As said before the planner used is the one given by the Professor, which uses the interpolation of a 5^{th} order polynomial for the acceleration (thus a 7^{th} order for the position). Once the desired position, velocity, acceleration, and the angle ψ are retrieved, using the property of the differential flatness for the quadrotor, it was easy to compute the other input necessary for the system. Also the results of the simulation are compared to the geometrical control ones, using the same set of parameters but optimized gains for itself. Here it is seen the output of this simulation:

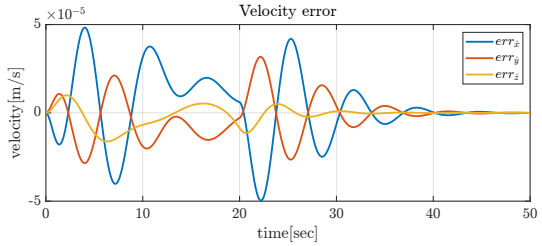
[2] The z-coordinate is negative cause the NED convention is adopted



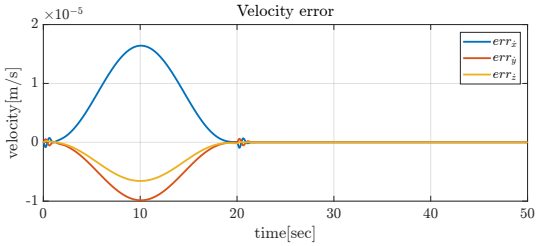
(a) position error hierarchical



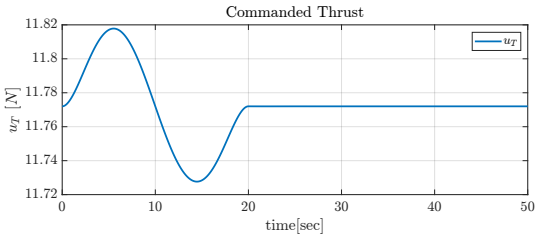
(b) position error geometrical



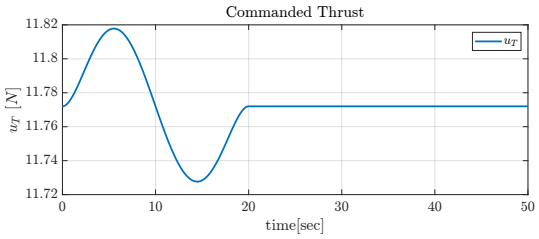
(c) velocity error hierarchical



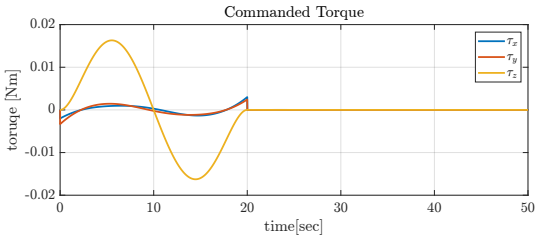
(d) velocity error geometrical



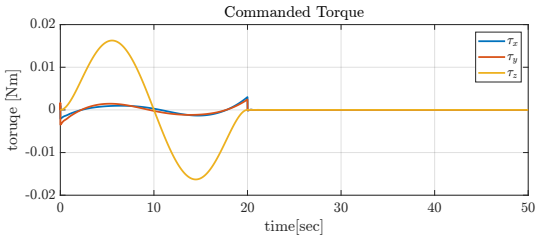
(e) Commanded thrust hierarchical



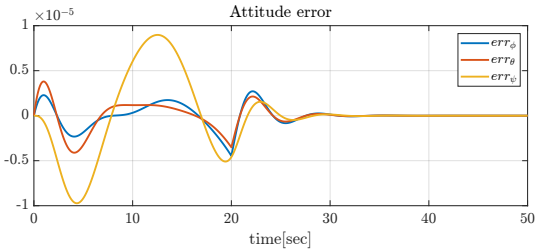
(f) Commanded thrust geometrical



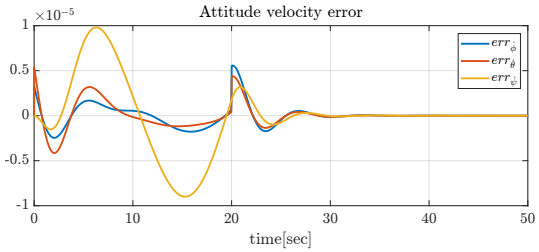
(g) Commanded torque hierarchical



(h) Commanded torque geometrical



(i) Attitude error hierarchical

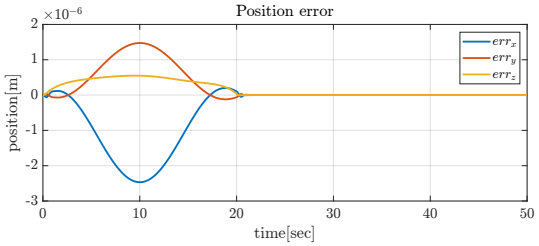


(j) Velocity attitude error hierarchical

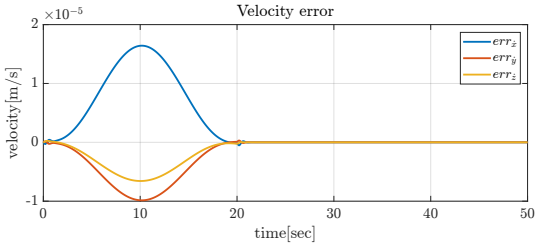
Figure 3: First simulation.

The first eight pictures compare the position error, the velocity error, the command thrust and the command torques of the hierarchical control (the one on the left) and the geometrical control (the one to the right). It's immediate to see how the error is totally different between the two; in the geometrical after 22s it goes to zero while in the hierarchical it goes to zero around 40s (both for the position and velocity). Nonetheless the error is of the order of 10^{-5} in the hierarchical, which is acceptable for this quadrotor; in the geometrical is two order lower but this is due to the gains given for the outer loop and inner loop. One may think to increase also the gains for the first controller, and this will be shown in the next simulation, observing also all the consequence it will bring.
Next it can be observed how the command thrust and the command torques are piratically identical to each other stopping at 20s cause it is the time given to reach the final point.
Lastly, the ninth and tenth pictures are the attitude error and velocity attitude error in the hierarchical control. There are not for the geometrical because the writing of the model in the coordinate free gives us nothing to compare with (can be retrieved the error on the rotational matrix or the angular velocity but are not comparable in the hierarchical). As shown, even these errors are of the order of 10^{-5} which is acceptable in this case; also it can be seen that at 20s there is a sudden change in the errors. This is due to the fact that the UAV reaches the desired point and when it arrives it tries to stop immediatly leading to some oscillation in the reality (like a spring that tries to return in the original position), but these sudden changes are so little that are imperceptible to the human eye.
Now, as anticipated, it will be performed a simulation increasing the gains of the controller, thus trying to achieve better performances. In particular everything else will be left untouched, while the gains are bumped to:

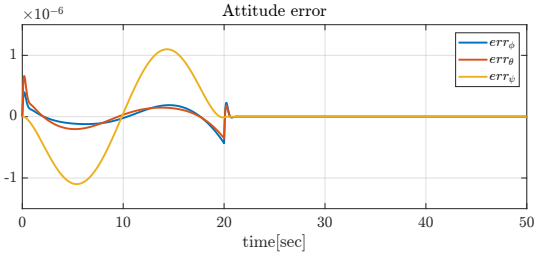
- $newKP = \begin{bmatrix} K_p & K_v \end{bmatrix}$ with $K_p = diag(\begin{bmatrix} 60 & 60 & 60 \end{bmatrix})$ and $K_v = diag(\begin{bmatrix} 8 & 8 & 8 \end{bmatrix})$ as the gains for the outer-loop controller
- $newKE = \begin{bmatrix} K_{etap} & K_{etav} \end{bmatrix}$ with $K_{etap} = diag(\begin{bmatrix} 60 & 60 & 60 \end{bmatrix})$ and $K_{etav} = diag(\begin{bmatrix} 10 & 10 & 10 \end{bmatrix})$ as the gains for the inner-loop controller



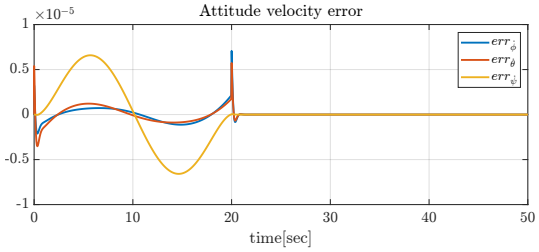
(a) position error hierarchical



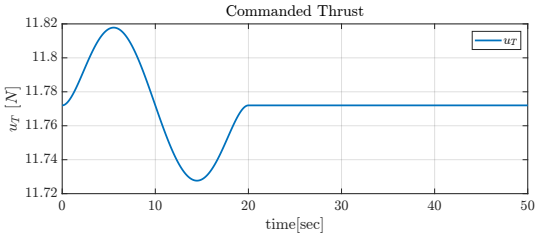
(b) velocity error hierarchical



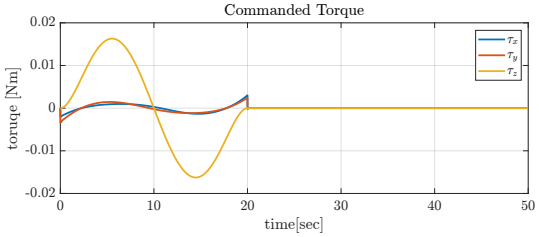
(c) Attitude error hierarchical



(d) Velocity attitude error hierarchical



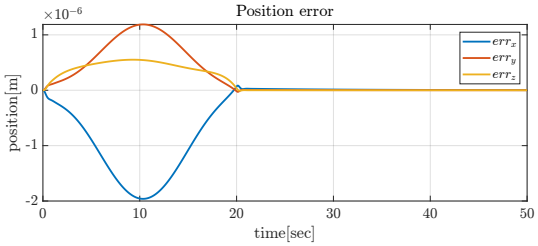
(e) Commanded thrust hierarchical



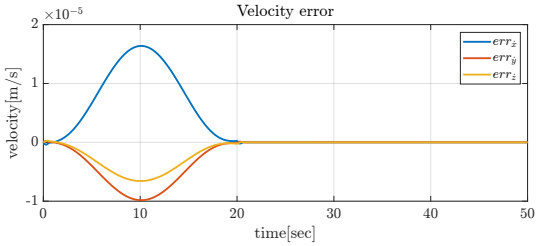
(f) Commanded torques hierarchical

Figure 4: Second simulation.

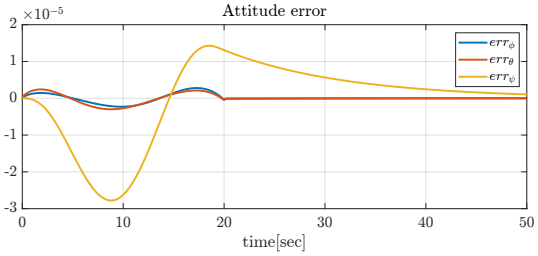
As seen, the bumped gains made the output practically identical, in the form and in the values, at the ones from the geometrical, thus the two controller are equivalent.
One last simulation was made with the modification of only one set of gains; in particular the gains in the inner loop of the attitude velocity error.



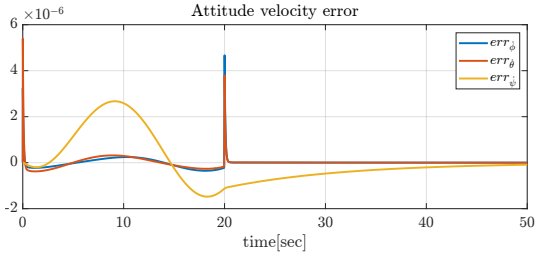
(a) position error hierarchical



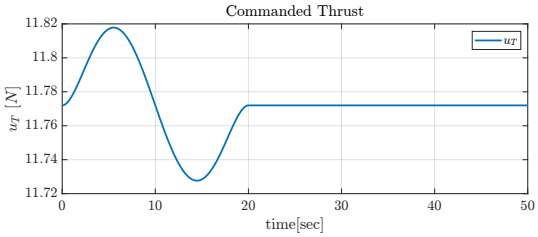
(b) velocity error hierarchical



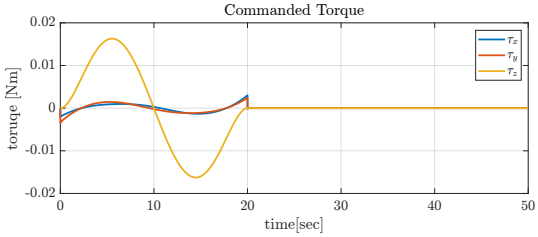
(c) Attitude error hierarchical



(d) Velocity attitude error hierarchical



(e) Commanded thrust hierarchical



(f) Commanded torques hierarchical

Figure 5: Third simulation.

Here not much is changed in the position and velocity error, or the command thrust and torques, but it is observed a significant change in the attitude error and attitude velocity error. In this case are not present anymore the oscillation seen previously, in fact it goes a lot more smooth to zero, but a lot more slower (it present also an higher peak in the attitude velocity error at 20s compared to before).

4 APF-algorithm

The second half of the elaborate it focuses on the implementation of an algorithm of obstacle avoidance. In particular it was chosen the artificial potential field (APF) algorithm. This algorithm is based on the fact that the planning does not require the knowledge of the path to the desired position but only the desired position is necessary. In fact it is not necessary even the knowledge of the obstacles occupancy in advance; thus the aim is to not build the C_{free}, but only to reach the final point.
The artificial potential method uses the concept of two simple potentials:

- an attractive potential to the goal
- a repulsive potential away from the CO-obstacle region

It is intuitive that the former is the potential necessary to guide the UAV to the goal, while the latter is the potential necessary to avoid collision (basically it creates a virtual barrier around the obstacle). From this potentials can be retrieved the two forces used for the planner in Simulink. Th expression of the first one is:

$$\begin{cases} f_a(q) = k_a e(q) & ||e(q)|| < 1 \\ f_a(q) = k_a \frac{e(q)}{||e(q)||} & ||e(q)|| \geq 1 \end{cases}$$

where:

- $k_a > 0$ is the attractive potential gain
- $e(q) = q_g - q$ is the error between the final position and the actual position

Note that the second term is useful in the case the initial error is a lot greater than expected, leading to a force so high that it would be impossible to realize in reality. The downgrade of using the norm is the fact that it goes a lot slower.
Meanwhile the second force can be expressed as:

$$\begin{cases} \frac{k_{r,i}}{\eta_i(q)^2} \left(\frac{1}{\eta_i(q)} - \frac{1}{\eta_{o,i}} \right)^{\gamma-1} \nabla \eta_i(q) & \eta_i(q) \leq \eta o, i \\ 0 & \eta_i(q) > \eta o, i \end{cases}$$

where:

- $k_{r,i} > 0$ is the repulsive potential gain
- $\eta_i(q) = \min_{q' \in CO_i} ||q - q'||$ is the distance from the obstacle i
- $\eta_{o,i}$ is the range of influence of the obstacle
- $\gamma = \{2, 3\}$

This force is calculate for every obstacle i, so in the end the total repulsive force is:

$$f_r(q) = \sum_{i=1}^{p} f_{r,i}(q)$$

Finally, the total force given to the quadrotor is the sum of the attractive and repulsive force:

$$f_t(q) = f_a(q) + f_r(q)$$

One last consideration before implementing this algorithm into the model of the UAV is to how considerate this force. There are three options:

- the total force is seen as the vector of generalized forces (force plus torque) that induce a motion in the robot.
- the total force is seen as the acceleration moving the robot.
- the total force is seen as the velocity vector moving the robot.

There are some unique advantages by choosing each of the options, but in this work the third one is selected, because it was simpler to generate trajectory offline.

The first simulation done was made by using the hierarchical control and the geometrical control and generating only one obstacle in the way of the motion of the robot. The obstacle is a cylinder positioned at $(8 \quad 0)$ with a radius of 2.5 and a height of 10. For simplification the starting position of the robot is at $\begin{bmatrix}1 & -1 & -5\end{bmatrix}$ and the final position at $\begin{bmatrix}19 & -1 & -5\end{bmatrix}$ Regarding the other parameters used, they are:

- $newKP = \begin{bmatrix}K_p & K_v\end{bmatrix}$ with $K_p = diag(\begin{bmatrix}30 & 30 & 30\end{bmatrix})$ and $K_v = diag(\begin{bmatrix}5 & 5 & 5\end{bmatrix})$ as the gains for the outer-loop hierarchical controller
- $newKE = \begin{bmatrix}K_{etap} & K_{etav}\end{bmatrix}$ with $K_{etap} = diag(\begin{bmatrix}30 & 30 & 30\end{bmatrix})$ and $K_{etav} = diag(\begin{bmatrix}7 & 7 & 7\end{bmatrix})$ as the gains for the inner-loop hierarchical
- $K_p = diag(\begin{bmatrix}100 & 100 & 100\end{bmatrix})$ and $K_v = diag(\begin{bmatrix}8 & 8 & 8\end{bmatrix})$ as the gains for the outer-loop geometrical controller
- $K_r = diag(\begin{bmatrix}20 & 20 & 3.5\end{bmatrix})$ and $K_w = diag(\begin{bmatrix}5 & 6.5 & 2.5\end{bmatrix})$ as the gains for the inner-loop geometrical controller
- $K_{attr} = 0.8$ and $K_{rep} = 0.6$ as the gains of attractive and repulsive potentials
- $\eta_{o,i} = 3.5$

Furthermore, once the data of the cylinder are defined, the function *points generation* is used to generate the set of points that represents the cylinder in the space. After this, the **UAV toolbox** is employed to generate a space and a mesh for the ground, the quadrotor, and the cylinder, for visualization purpose. The following picture show the simulink scheme for the new planner:

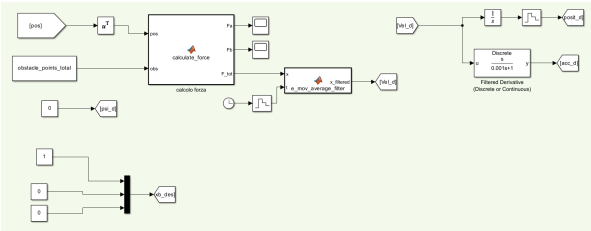


Figure 6: Simulink scheme for apf algorithm

In this scheme, are present, in addition to the calculation of the forces, and consequently the velocity, acceleration and position given to the robot, the angle ψ desired and the x_b desired. The first one is used in the hierarchical control, while the second one for the geometrical. Also the implementation of a exponential moving average filter is used at the exit of the block **calculate force** cause otherwise the UAV at the start it would behave in a more 'acrobatic' way than expected. Here we see the trajectory of the quadrotor, as well the output generated:

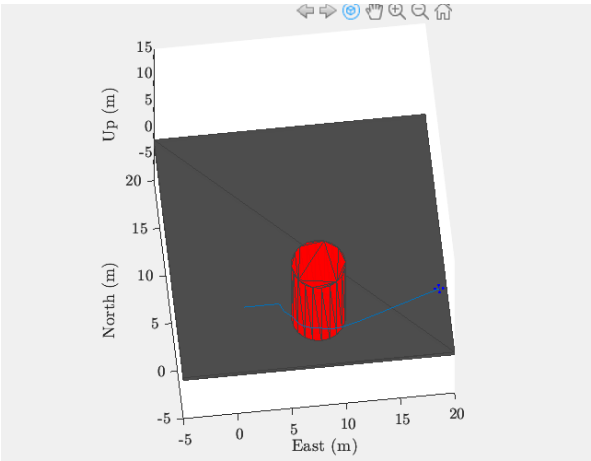
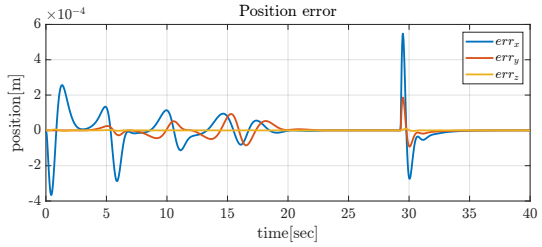
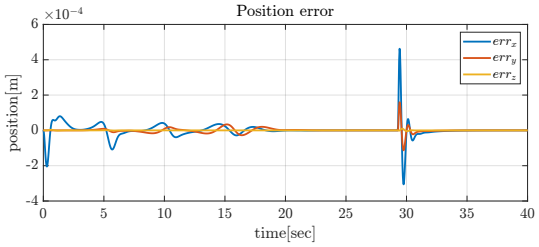


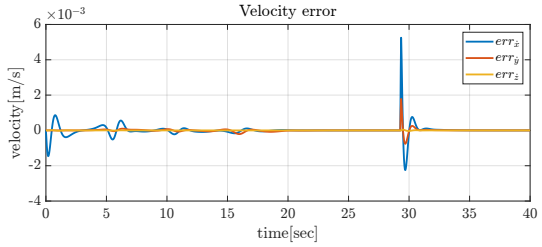
Figure 7: Trajectory of the UAV using the APF in the hierarchical control



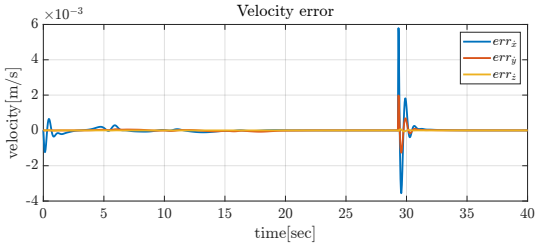
(a) position error hierarchical



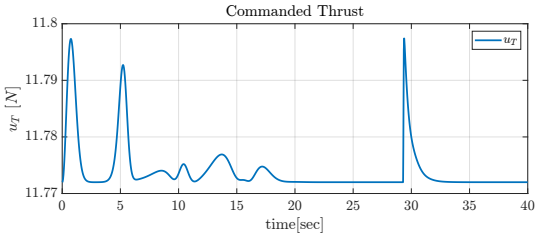
(b) position error geometrical



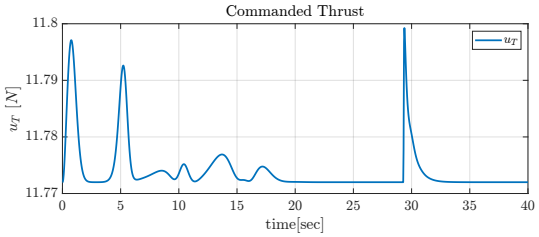
(c) velocity error hierarchical



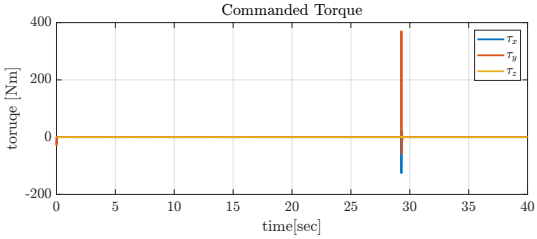
(d) velocity error geometrical



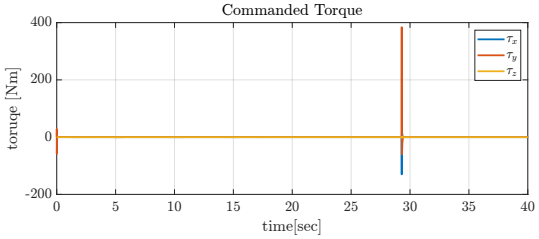
(e) Commanded thrust hierarchical



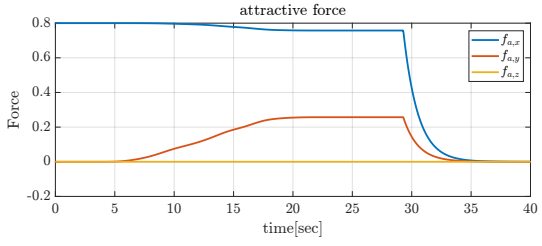
(f) Commanded thrust geometrical



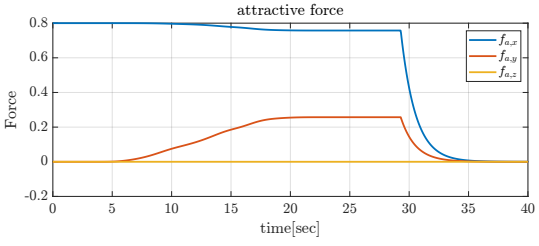
(g) Commanded torque hierarchical



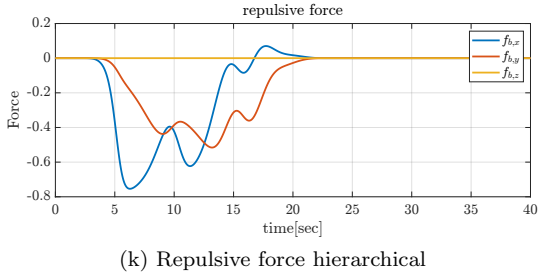
(h) Commanded torque geometrical



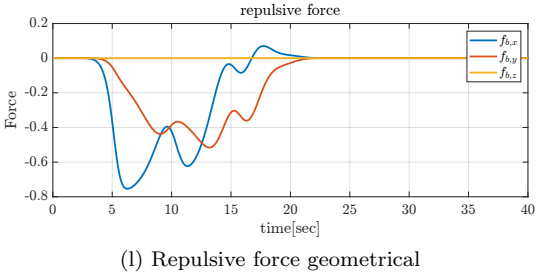
(i) Attractive force hierarchical



(j) Attractive force geometrical



(k) Repulsive force hierarchical



(l) Repulsive force geometrical

Figure 8: APF simulation.

As seen from the first picture the robot notices the obstacle and the repulsive force is different from zero, 'pushing' to the side the UAV. Once it is 'pushed' enough (meaning there is not anymore the contribute from the repulsive force), the UAV goes to the desired point in a straight line.
The second set of images are the outputs of the simulation. To the left the hierarchical, to the right the geometrical. The set of graph are practically identical to each other (in fact from now on the simulation will be made only on the hierarchical control); there are only some difference in the position and velocity error but they are neglectable because we are in the order of $^{-3}$. From the graph of the force, it is shown that the obstacle it is seen around 5s, in fact before this only the force along x was present. After 5s the repulsive forces start to appear and the attractive force along y get in the game. Around 20s the repulsive forces become zero, meaning that the UAV as encircled the cylinder, in fact on the other end the attractive forces are returned to be constant until the final point is reached (around 30s). The spike at the end it is due to the fact that the forces are not differentiable in that point.

5 Local minima problem

One of the problem that the artificial potential method suffer is the possibility to incur in a local minima.
A local minima is a point in which the total potential is not zero (the potential is only zero if it is on the desired point), but the total force is equal to zero.
This can occur, for example, when the UAV encounters a wall between itself and the goal, thus the repulsive force and the attractive force are equal in absolute value but with opposite directions. The following pictures show the simulation that achieved this. Basically it starts with the same set of parameters as the normal APF, but the only difference is that the cylinder is centered at $\begin{bmatrix} 8 & -1 \end{bmatrix}$ instead at $\begin{bmatrix} 8 & 0 \end{bmatrix}$. Everything else is left unchanged

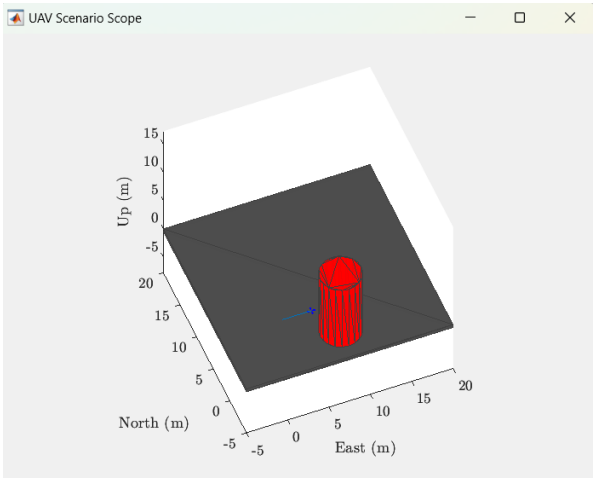
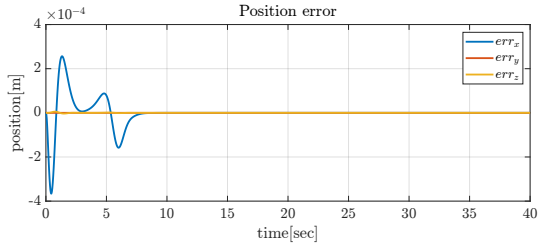
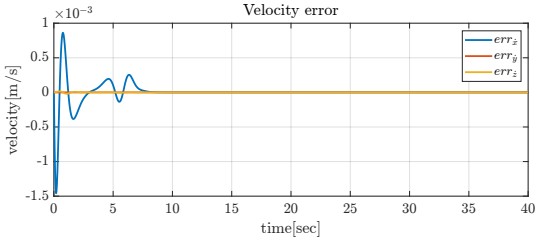


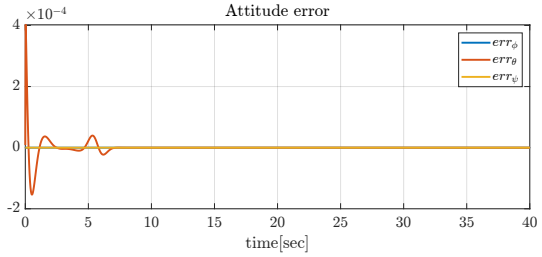
Figure 9: Trajectory of the UAV when discover a local minima using the APF in the hierarchical control



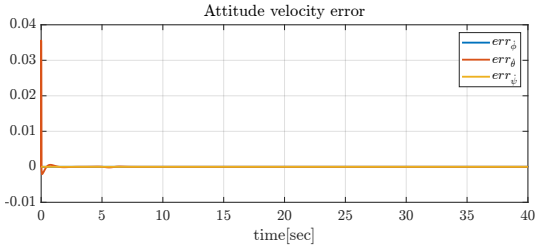
(a) position error hierarchical



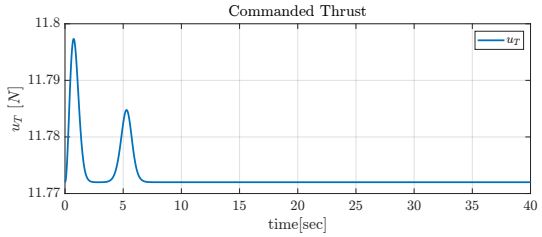
(b) velocity error hierarchical



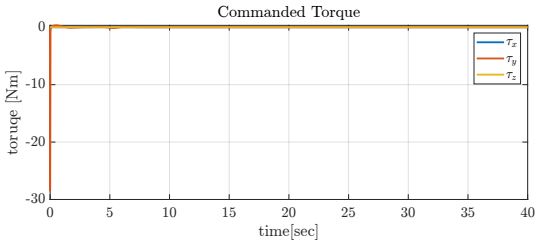
(c) attitude error hierarchical



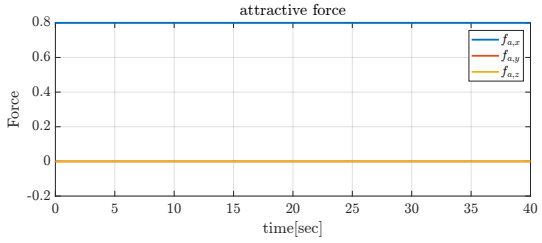
(d) velocity attitude error hierarchical



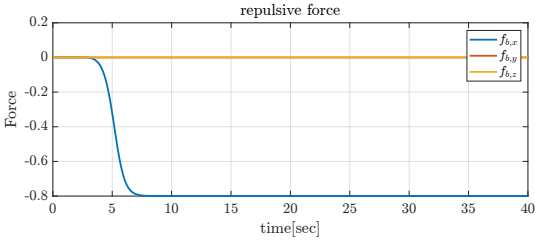
(e) command thrust hierarchical



(f) command torque hierarchical



(g) attractive force hierarchical



(h) repulsive force hierarchical

Figure 10: Local minima simulation.

As seen in the pictures, it immediately jumps at the eye that when the quadrotor discovers the obstacle (around 5s) the repulsive force becomes the same as the attractive force but with opposite sign, thus the robot is stationary. This is achieved because the intrinsic symmetric geometry of the cylinder and the fact that the quadrotor starts at that specif position (basically every component of the forces nullify each other).
Sadly, to resolve this problem, it must be taken action online. Basically the APF must be shut down online, then move the robot in a random way (always being careful to the environment and the obstacles) and in the end reactivate the APF algorithm (this is not presented in the work).

6 Conclusion

In this project was shown the implementation of a hierarchical control and the artificial potential field algorithm. In the first half was described how it works a hierarchical control and it was tested with different gains such control. The result are also confronted to the geometrical control to the same model. In the end, with appropriate gains the two control are virtually equivalent to each other.
The second half of the elaborate it was focused on to the APF algorithm. It was explained how this algorithm works and how it was implemented using Matlab/Simulink. In its entirety, was shown also the generation of a virtual map that the robot should navigate and the obstacles that should avoid. It was analyzed the result obtained first with the hierarchical and later with the geometrical, discovering that even in this case are identical; in fact the forces displayed (attractive and repulsive) are basically the same. In the end it was shown one of the flaw of the APF algorithm, which is the occurrence of local minima. In this specific point the potential of the robot is greater than zero, but the forces are equal to zero making the UAV stucked in place.