Attitude and Position Control of a Quadcopter in a Networked Distributed System

Alejandro Alonso García
Department of Electronic Systems
Control and Automation
Aalborg University
Email: aalons16@student.aau.dk

Amalie V. Petersen
Department of Electronic Systems
Control and Automation
Aalborg University
Email: apet13@student.aau.dk

Andrea Victoria Tram Løvemærke
Department of Electronic Systems
Control and Automation
Aalborg University
Email: alavem13@student.aau.dk

Niels Skov Vestergaard
Department of Electronic Systems
Control and Automation
Aalborg University
Email: nveste12@student.aau.dk

Noelia Villarmarzo Arruñada
Department of Electronic Systems
Control and Automation
Aalborg University
Email: nvilla16@student.aau.dk

***Abstract*—Quadcopters are becoming increasingly interesting due to the great variety of usage. A design that is able to make the quadcopter hover and move to a desired position is presented. The system's coupled behavior and instability raises a challenging control task. This task is solved by implementing a controller design, which is based upon a model that is derived by first principle physics. This is later linearized since it is desired to use linear controllers. The system is divided into an attitude and translational control loops. These are designed as state space and classical control, respectively. The prototype gets its attitude and position from a motion tracking system, keeping the control in a micro processor on the quadcopter. This layout constitutes a distributed system, where network issues, such as delays and packet losses, are taken into account.**

I. Introduction

In the last years, the interest for quadcopters has increased due to the great possibilities they offer. Among these, the most well-known ones are surveillance, inspection of big structures and search and rescue missions in difficult environments. **droneuses**

The quadcopter constitutes a control challenge due to its unstable nature and coupled behavior. The system has six degrees of freedom, the three position coordinates and the three orientations, and there are only four actuation variables, namely the motor rotational speeds. The dimension of the problem is explained by McKerrow in **draganflyer**

The control of a quadcopter has been addressed many times in the recent years. In Mian et al. **backstepping** the quadcopter is controlled using a back-stepping technique and non-linear controllers. Another way of solving the issue is presented in Tayebi et al. **quaternionsPD** in which the quadcopter attitude is modeled using quaternions and controlled with a PD based controller. In **MianWang** Mian and Wang model the system using its dynamic equations and use non linear controllers to achieve a steady flight while in Mokhtari et al. **GHinf** the system is controlled by a mixture of a robust feedback linearization and a modified optimization control method.

The approach presented here models the quadcopter by a first principles method. This approach yields a non linear model that describes the attitude and translational behavior of the quadcopter. The model is then linearized around an equilibrium point, which is chosen to be in hovering position. With the linearized equations, controllers for attitude and translational behaviors are designed. The attitude controller is obtained by means of a state space representation while the translational controller is designed using classical control. In the control system, the translational constitutes an outer loop and sets the reference for the attitude controller. Since the sensors are not placed in the quadcopter and the information comes from an external motion tracking system **vicon** an analysis on how the network affects the control loop is also presented. Lastly, the simulations and experimental results of the designed controllers are presented and discussed.

II. Model

The quadcopter free body diagram is shown in Figure 1.

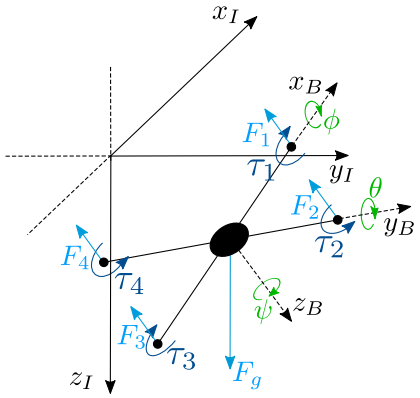

Figure 1: Forces and torques acting on the quadcopter and the positive references chosen for rotations and translations in both inertial and body coordinate frames.

As it is seen, the system is modeled by using two coordinate frames. The inertial frame is utilized to describe the translational movement while the body frame is attached to the quadcopter and used to characterize its attitude behavior. In the figure, also the positive references for rotational and translational movements are depicted, as well as the main forces and torques acting on the quadcopter.

The forces generated in the propeller are readily obtained in the body coordinate frame. In order to represent them in the inertial frame a rotation matrix is used. It is built considering a 123 rotation sequence **rotationmatrix**

The dynamic model of the quadcopter is given by through three sets of equations. The first describes the motor and the propeller, the second presents the attitude response of the quadcopter and the third explains how the translational variables of the system evolve.

A. Motor and Propeller

The four motors in the quadcopter generate a rotation in the propellers that creates the force that lifts the quadcopter. The thrust force can be modeled as proportional to the square of the motor rotational speed. The thrust coefficient is found experimentally.

The rotation also generates a torque on each motor due to drag between air and propeller. Drag torque is compensated in the quadcopter by having two of the motors turning in one direction and the two others in the opposite direction. It is as well described as proportional to the square of the rotational speed in terms of a drag coefficient, which is also obtained experimentally.

Equation 1 and 2 show the expression for the thrust force and drag torque caused by the rotation of each propeller.

$$F = k_{th}\omega^2 \tag{1}$$

$$\tau = k_d\omega^2 \tag{2}$$

Where F is the thrust force, k_{th} is the thrust coefficient, ω is the angular speed of the motor, τ is the drag torque and k_d is the drag coefficient.

These equations are used in the attitude and translational models presented below.

B. Attitude Model

The attitude model equations are based on Newton's Second Law for rotational movement and are represented in Equation 3, 4 and 5.

$$J_x\ddot{\phi} = k_{th}(\omega_4^2 - \omega_2^2)L \tag{3}$$

$$J_y\ddot{\theta} = k_{th}(\omega_1^2 - \omega_3^2)L \tag{4}$$

$$J_z\ddot{\psi} = k_d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \tag{5}$$

Where J_x, J_y and J_z are the moments of inertia around the three axes of rotation, $\ddot{\phi}$, $\ddot{\theta}$ and $\ddot{\psi}$ are the accelerations in roll, pitch and yaw angles, respectively, ω_i is the rotational speed of each motor and L is the distance between the center of the quadcopter and the position of the motors.

The expressions above state how the thrust forces and the drag torques generated on the propellers affect the attitude behavior of the quadcopter.

C. Translational Model

The equations describing the response of the system along the inertial x, y and z axes are derived from Newton's Second Law. The forces that act on the system are those from the propellers and the gravitational force. These expressions are shown in Equation 7, 8 and II-C.

$$\begin{aligned} m\ \ddot{x}_I =& -k_{th}\ ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) \\ & \cdot (\cos(\phi)\ \sin(\theta)\ \cos(\psi) + \sin(\phi)\ sin(\psi)) \end{aligned} \tag{6}$$

$$\begin{aligned} m\ \ddot{y}_I =& -k_{th}\ ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) \\ & \cdot (\cos(\phi)\ \sin(\theta)\ \sin(\psi) - \sin(\phi)\ \cos(\psi)) \end{aligned} \tag{7}$$

$$\begin{aligned} m\ \ddot{z}_I =& F_g - k_{th}\ ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) \\ & \cdot \cos(\phi)\ \cos(\theta) \end{aligned} \tag{8}$$

Where m is the mass of the quadcopter, $\ddot{x}_I$, $\ddot{y}_I$ and $\ddot{z}_I$ are the accelerations along the inertial reference frame directions, ϕ, θ and ψ are the roll, pitch and yaw angles respectively and F_g is the gravitational force acting on the quadcopter.

It is worth mentioning that, as the thrust forces always point in the negative z_B direction, the accelerations along x_I and y_I directions are zero when pitch and roll angles are zero.

D. Linearization

The model equations are linearized using the first order Taylor approximation around an equilibrium point of the system. The chosen point is the hovering position, in which all state derivatives have a value of zero, that is, the attitude and translational accelerations and velocities. The angular position of the quadcopter is also part of point around which the linearization is made.

Choosing a zero acceleration linearization point along the z_I axis yields a linearization motor rotational speed so that the necessary thrust is generated to compensate for the gravitational force. Equation 9 expresses this relation.

$$\overline{\omega}_i = \sqrt{\frac{m\ g}{4k_{th}}} \tag{9}$$

The resulting equations for the attitude model after the linearization are shown in Equation 10, 11 and 12.

$$J_x\ \Delta\ddot{\phi} = 2\ k_{th}\ L\ \overline{\omega}_4\ \Delta\omega_4\ -\ 2\ k_{th}\ L\ \overline{\omega}_2\ \Delta\omega_2 \tag{10}$$

$$J_y\ \Delta\ddot{\theta} = 2\ k_{th}\ L\ \overline{\omega}_1\ \Delta\omega_1\ -\ 2\ k_{th}\ L\ \overline{\omega}_3\ \Delta\omega_3 \tag{11}$$

$$\begin{aligned} J_z\ \Delta\ddot{\psi} =& 2\ k_d\ \overline{\omega}_1\ \Delta\omega_1\ -\ 2\ k_d\ \overline{\omega}_2\ \Delta\omega_2 \\ & +\ 2\ k_d\ \overline{\omega}_3\ \Delta\omega_3\ -\ 2\ k_d\ \overline{\omega}_4\ \Delta\omega_4 \end{aligned} \tag{12}$$

Where $\Delta\ddot{\phi}$, $\Delta\ddot{\theta}$ and $\Delta\ddot{\psi}$ are the changes in rotational acceleration from the linearization point, $\overline{\omega}_i$ is the rotational speed of each motor in to achieve equilibrium along the z_I axis and $\Delta\omega_i$ is the change in rotational speed of each motor from the linearization point.

Similarly, the equations of the translational model are linearized. The result is shown in Equation 13, 14 and 15.

$$m\ \Delta\ddot{x}_I = -k_{th}\ (\overline{\omega}_1{}^2 + \overline{\omega}_2{}^2 + \overline{\omega}_3{}^2 + \overline{\omega}_4{}^2)\ \Delta\theta \tag{13}$$

$$m\ \Delta\ddot{y}_I = k_{th}\ (\overline{\omega}_1{}^2 + \overline{\omega}_2{}^2 + \overline{\omega}_3{}^2 + \overline{\omega}_4{}^2)\ \Delta\phi \tag{14}$$

$$m\ \Delta\ddot{z}_I = -2\ k_{th}\ \overline{\omega}_1\ \Delta\omega_1 - 2\ k_{th}\ \overline{\omega}_2\ \Delta\omega_2 - 2\ k_{th}\ \overline{\omega}_3\ \Delta\omega_3 - 2\ k_{th}\ \overline{\omega}_4\ \Delta\omega_4 \tag{15}$$

Where $\Delta\ddot{x}_I$, $\Delta\ddot{y}_I$ and $\Delta\ddot{z}_I$ are the changes in linear acceleration from the linearization point in each direction of the inertial frame and $\Delta\phi$ and $\Delta\theta$ are the changes in roll and pitch from the linearization point, respectively.

III. Control

The control of the system is divided into two control systems. One handles the attitude and the other controls the translational behavior of the quadcopter. These two are related such that the translational controller sets the references for the angles handled by the inner controller.

A. Attitude Controller

The attitude controller for the quadcopter is designed using a state space representation of the system. This helps handling the coupled angular response of the quadcopter. The chosen states for the system are the three angular positions and the three angular velocities. The input vector of the attitude system consists of the four motor rotational speeds and the output vector consists of the three angles, roll, pitch and yaw. Below, the state, input and output vectors are presented.

$$\mathbf{x}(t) = \begin{bmatrix}\phi & \theta & \psi & \dot{\phi} & \dot{\theta} & \dot{\psi}\end{bmatrix}^T$$

$$\mathbf{y}(t) = \begin{bmatrix}\phi & \theta & \psi\end{bmatrix}^T$$

$$\mathbf{u}(t) = \begin{bmatrix}\omega_1 & \omega_2 & \omega_3 & \omega_4\end{bmatrix}^T$$

The vectors above are then used in construction of the state space matrix representation as displayed in Equation 16 and 17.

$$\dot{\mathbf{x}}(t) = \mathbf{A}\cdot\mathbf{x}(t) + \mathbf{B}\cdot\mathbf{u}(t) \tag{16}$$

$$\mathbf{y}(t) = \mathbf{C}\cdot\mathbf{x}(t) + \mathbf{D}\cdot\mathbf{u}(t) \tag{17}$$

Where $\mathbf{A}$ is the system matrix, $\mathbf{B}$ is the input matrix, $\mathbf{C}$ is the output matrix and $\mathbf{D}$ is the feedback matrix.

The values for the $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ matrices are obtained from the linearized attitude equations, yielding the matrices shown below. As $\mathbf{D}$ is a zero matrix, only $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are shown.

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{2\cdot k_{th}\cdot L\cdot\overline{\omega}_2}{J_x} & 0 & \frac{2\cdot k_{th}\cdot L\cdot\overline{\omega}_4}{J_x} \\ \frac{2\cdot k_{th}\cdot L\cdot\overline{\omega}_1}{J_y} & 0 & -\frac{2\cdot k_{th}\cdot L\cdot\overline{\omega}_3}{J_y} & 0 \\ \frac{2\cdot k_d\cdot\overline{\omega}_1}{J_z} & -\frac{2\cdot k_d\cdot\overline{\omega}_2}{J_z} & \frac{2\cdot k_d\cdot\overline{\omega}_3}{J_z} & -\frac{2\cdot k_d\cdot\overline{\omega}_4}{J_z} \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

The attitude control is based on a state feedback and an integral term in order to be able to track a given reference. As not all states are measured a reduced order observer is implemented to estimate the angular velocities of the quadcopter. Due to the separation principle, both subsystems can be designed independently. **ssReference**

Figure 2 shows how these designs are related.

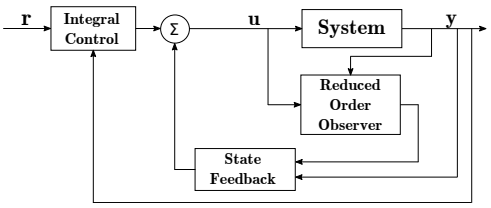

Figure 2: Control structure for the system, including the state feedback, the integral controller and the reduced order observer.

The design of the state feedback and integral control is shown in Figure 3.

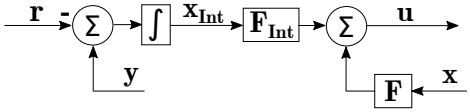

Figure 3: State feedback and integral controller in the attitude control structure.

As there are three outputs to track, three states, $\mathbf{x}_{int}(t)$, are added to the already existing state vector. This leads to the extended system shown in Equation 18 and 19.

$$\dot{\mathbf{x}}_e(t) = \mathbf{A}_e\ \mathbf{x}_e(t) + \mathbf{B}_e\ \mathbf{u}(t) + \begin{bmatrix}\mathbf{0}\\ -\mathbf{I}\end{bmatrix}\mathbf{r}(t) \tag{18}$$

$$\mathbf{y}(t) = \mathbf{C}_e\ \mathbf{x}_e \tag{19}$$

Where

$$\dot{\mathbf{x}}_e(t) = \begin{bmatrix}\dot{\mathbf{x}}(t)\\ \dot{\mathbf{x}}_{int}(t)\end{bmatrix} \quad \mathbf{A}_e = \begin{bmatrix}\mathbf{A} & \mathbf{0}\\ \mathbf{C} & \mathbf{0}\end{bmatrix} \quad \mathbf{B}_e = \begin{bmatrix}\mathbf{B}\\ \mathbf{0}\end{bmatrix} \quad \mathbf{C}_e = \begin{bmatrix}\mathbf{C} & \mathbf{0}\end{bmatrix}$$

The feedback law for this design approach is given by Equation 20. Its design is done as a conventional state feedback, where the goal is to choose an appropriate $\mathbf{F}_e = [\mathbf{F}\ \mathbf{F}_{int}]$

matrix, such that the eigenvalues of $\mathbf{A}_e + \mathbf{B}_e\mathbf{F}_e$ are the closed loop poles giving the desired dynamics.

$$\mathbf{u}(t) = \mathbf{F}\ \mathbf{x}(t) + \ \mathbf{F}_{int}\ \mathbf{x}_{int}(t) \tag{20}$$

Once $\mathbf{F}_e$ is obtained, it is split into F and F_{int}. In this way, the controller is implemented as shown in Figure 3.

Since certain states in the system are not measured, an observer estimates the remaining states by means of the system output and input. With this approach, the first three states, $\mathbf{x}_1$, are equal to the outputs, $\mathbf{y}$, whereas the other three states, $\mathbf{x}_2$, are estimated as $\hat{\mathbf{x}}_2$.

The observer is designed by finding the matrix $\mathbf{L}_{obs}$ such that the eigenvalues of the matrix $\mathbf{A}_{22} + \mathbf{L}_{obs}\ \mathbf{A}_{12}$ **ssReference** This is done splitting the original system matrices into sub-matrices, as seen below.

$$\mathbf{A} = \begin{bmatrix} \mathbf{A11} & \mathbf{A12} \\ \mathbf{A21} & \mathbf{A22} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} \mathbf{B1} \\ \mathbf{B2} \end{bmatrix}$$

With the observer matrix, the observer equation is derived. See Equation 21. This ensures an estimate $\hat{\mathbf{x}}_2$ which converges to $\mathbf{x}_2$ at a rate given by the chosen observer poles.

$$\dot{\hat{\mathbf{x}}}_\mathbf{2} = \mathbf{A_{21}y} + \mathbf{A_{22}}\hat{\mathbf{x}}_\mathbf{2} + \mathbf{B_2u} + \mathbf{L_{obs}}(\mathbf{A_{12}}\hat{\mathbf{x}}_\mathbf{2} - \mathbf{A_{21}x_2}) \tag{21}$$

This estimation of $\hat{\mathbf{x}}_2$ is also seen in Figure 4.

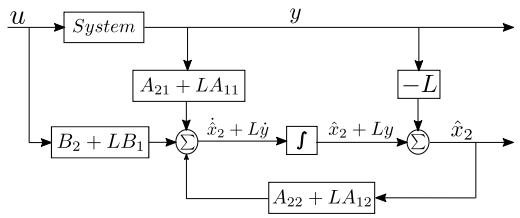

Figure 4: Detailed diagram of the reduced order observer that shows its implementation.

B. Translational Controller

The translational controllers are structured as cascade loops, where the velocity and position are controlled in the inner and outer loop, respectively. The relation between the controllers is presented in Figure 5.

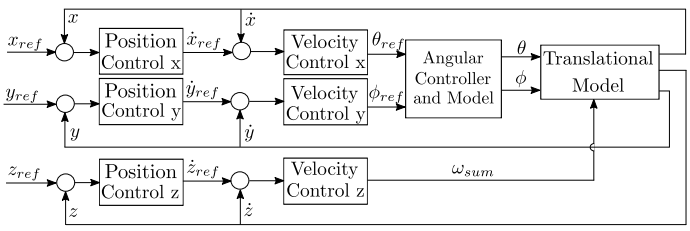

Figure 5: Overview of translational controllers structure.

The x and y controllers share similar properties as both their outputs are angle references, θ_{ref} and ϕ_{ref}, while the output of the z controller is the required sum of motor rotational speeds.

To design the inner controllers for the velocities $\dot{x}$ and $\dot{y}$, the model equations derived previously, see Equation 7 and 8, are Laplace transformed. These are used in Equation 22 and 23 to create a transfer function between the angles and the velocities, yielding:

$$G_{\dot{x}}(s) = \frac{\dot{x}(s)}{\theta(s)} = \frac{-k_{th}(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2)}{m\ s} \tag{22}$$

$$G_{\dot{y}}(s) = \frac{\dot{y}(s)}{\phi(s)} = \frac{k_{th}(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2)}{m\ s} \tag{23}$$

Where $G_{\dot{x}}$ and $G_{\dot{y}}$ are the plants used to design the velocity controllers in x_I and y_I directions respectively.

Since the plants are the same but with different signs, the controller design is carried out for $\dot{x}$ and applied to $\dot{y}$ afterwards with negative sign.

A proportional controller is chosen as the plant already has an integrator, which eliminates steady state tracking error and the effect of output disturbances. The gain is designed such that the system has a bandwidth that is three times lower than the attitude control loop to ensure that its dynamics do not affect the designed controller.

The outer loop is again designed to have three times less bandwidth than the inner velocity loop. Then, the plant of the outer loop is only an integrator that transforms velocity to position. As for the inner loop, the controller of the outer loop is a proportional controller.

To be able to design the inner loop for the z translational controller, the model equation derived previously, see Equation II-C, is Laplace transformed and written on the form of a transfer function, as seen in Equation 24.

$$G_{\dot{z}} = \frac{\dot{z}}{\omega_{sum}} = \frac{\frac{1}{4}\ (-2k_{th})\ \overline{\omega}_{sum}}{m\ s} \tag{24}$$

Where $\dot{z}$ is the velocity in the z_I direction, ω_{sum} is the sum of the rotational speeds of the motors and $\overline{\omega}_{sum}$ is the sum of the rotational speeds in equilibrium.

Due to an integrator and a negative gain the system's root locus will move into the right half plane as the gain increases. A proportional controller with negative gain ensures the system to be stable.

As there is not any inner control loop, input disturbances may occur such that a steady state error appears, which is not eliminated by the integrator in the plant. To remove this potential error an integrator is added to the controller.

The plant for the z position controller is just an integrator and a proportional controller is utilized. This is as well designed so that the bandwidth frequency is three times lower than the inner one.

IV. Network

The wireless network between the sensor and the quadcopter can influence the performance of the controller. This influence can be divided into two categories: the delay, from when the

sensor receives data until the microcontroller utilizes the data, and the loss of packages in the communication channel.

The theoretical modeling of these influences has been studied by several researchers in order to find a maximum allowable delay and packet loss, while the control system remains stable **wang walsh** However, this approach often leads to an increased complexity in the model.

An alternative to account for these effects in the control system is to utilize a network simulator such as TrueTime **TrueTimeNew**

TrueTime enables the opportunity to simulate the network and the controller together. It is thereby possible to find the packet loss probability and the maximum occurring delay in the system, while ensuring the control system is still stable. This is then compared to the delay occurring in reality. The delay is modeled as an exponential distribution. Its mean parameter is obtained experimentally by averaging multiple delay measurements in the communication channel. The packet loss, defined as a constant probability of loosing a packet, is found experimentally by sending a large amount of packages and examine how many of these are lost.

V. Results

The controllers have been simulated in MATLAB Simulink in order to generate results presented in this section.

The model parameters can be seen in Table I. All of them have been obtained through test except for the moments of inertia, which were calculated by following an analytical procedure.

Symbol	Value	Units
m	0.996	kg
L	0.225	m
J_x	0.01073	$kg\ m^2$
J_y	0.01073	$kg\ m^2$
J_z	0.02135	$kg\ m^2$
k_{th}	1.32922 10^{-5}	$N\ s^2\ rad^{-2}$
k_d	9.39741 10^{-7}	$N\ m\ s^2\ rad^{-2}$
$\overline{\omega}_i$	429	$rad\ s^{-1}$

Table I: Parameters used though the analysis and design.

The attitude controller is defined by the chosen feedback, integral poles, $[-6-6.2-6.4-6.6-6.8-7-7.2-7.4-7.6]$, and the observer poles, $[-20-25-30]$.

The translation velocity controllers for x and y are $C_{\dot{x}}(s) = -0.1$, $C_{\dot{y}}(s) = 0.1$ and the position ones are $C_x(s) = 0.5$, $C_y(s) = 0.5$. The PI-controller for the z translational velocity is $C_{\dot{z}}(s) = \frac{-201s+0.8}{s}$ and the outer loop P-controller is $C_z = 0.9$.

These controllers are discretized using the Tustin method with a sampling period of 30 ms.

Regarding the network, the mean delay selected is 30 ms and the package loss probability is set to 0 due to the low sending frequency used in the system. The simulation results obtained with these parameters and controllers are shown in Figure 6.

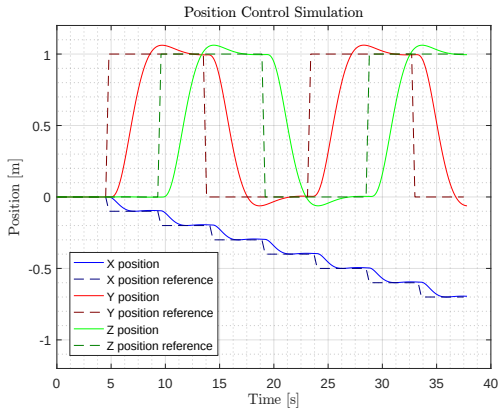

Figure 6: Position control results in the three inertial axes directions. The references given to the control system are shown with dashed lines.

The inner attitude controller results are also included and shown in Figure 7 so the performance of the attitude control can be evaluated.

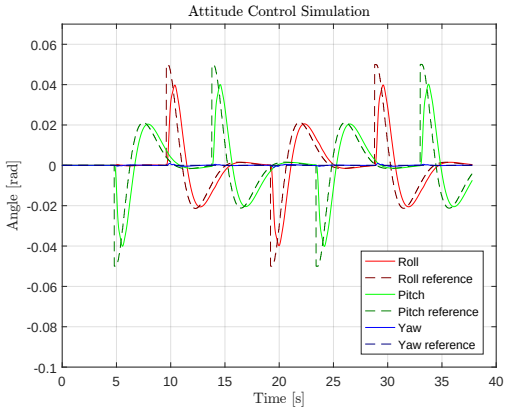

Figure 7: Attitude control results in the three angles. The references given to the attitude control system are shown with dashed lines.

VI. Discussion

The results obtained in the simulations show both the attitude and position response of the quadcopter.

It is seen that the controllers achieve the desired reference even though the network delay and the sampling rate affect the performance. The main network effect is the designed bandwidth of the controllers. This occurs due to the limited frequency in which the sensor data is obtained from the motion tracking system through the wireless connection.

It is also worth observing how the attitude controller shows a permanent error with respect to the reference. This is generated as a result of the integral controller design because it assumes a constant reference applied to it. This issue, though, does not affect the final position of the quadcopter.

VII. Conclusion

The behavior of a quadcopter has been modeled by first principles of physics. A control system has been designed in order to hover and move to a desired position. The control system has been split into an attitude and a translational controller. The first one has been designed using a state space approach, including state feedback with integral control and a reduced order observer. The translational control system has been designed with a classical control approach and result in three cascade loops, including proportional and PI controllers. As the quadcopter uses an external motion tracking system to determine its position and orientation, an analysis of the issues that can arise when having a networked distributed system has been done in order to ensure the control system remains stable. The results obtained from the design show that both the attitude and the translational behavior of the quadcopter has been successfully controlled.

VIII. Future Work

The control system has been designed using a motion tracking system as an attitude sensor. In order to improve the result while keeping the same control structure, an inertial measurement unit could be installed on the quadcopter. [1]

[1] FiXme Note: Acknowledgements