
INTERNSHIP REPORT

# Flight Control and Navigation of a Quadcopter

Ana CASTILLO BENITO

Poitiers, 2013 - 2014

Advisors: Emmanuel GROLLEAU and Patrick COIRAULT

# Abstract

The objective of this study is the development of some control strategies to solve the path following tracking problem in an unmanned aerial vehicle as a quadcopter. For this purpose, a model of quadcopter has been proposed. It is base on physical laws to represent the behavior of a mini drone previously built in the laboratory.

This project is organized as follows:

- **Chapter 1** presents an introduction of the Unmanned Air Vehicles (UAV). A description of their characteristics is provided, as well as its introduction into the aviation world. Some of their multiple possible uses are described but also the obstacles that must be overcome before introducing them in the airspace. The chapter ends with some of the latest drones that are being developed by different companies.

- **Chapter 2** shows the modeling of the quadrotor. The main advantages and drawbacks of the vehicle as well as its operations are described. The equations of motion of the quadcopter are obtained through the Newton-Euler approach and the Euler-Lagrange approach. Also, a Simulink model has been built to prove that the equations represent the proper operation of the UAV. To conclude, the parameters of the quadcopter used in this work are provided.

- **Chapter 3** deals with different control strategies to perform the path tracking of the quadcopter. Three control structures are developed. The first one is a PID controller able to control the attitude and altitude of the vehicle. Different perturbations are introduced in the model to see how the system reacts. Secondly, an Integral Backstepping controller is introduced which allows the vehicle to follow the desired trajectory in the three axes. Two different inputs are simulated, the first one is a squared at constant altitude and the second one is an ascending helical path. The last controller is formed by an integral nonlinear controller in the inner loop and a model predictive controller in the outer loop. The same helical path as in the first controller is selected to be simulated. Moreover, a trajectory generator has been designed. Its objective is to select a trajectory when only the initial and final desired points are given.

- **Chapter 4** summarizes the results presented in this project and suggests possible futures works.

# Contents

# List of figures

**Chapter 1**



**Chapter 2**

## Chapter 3

# List of tables

# Chapter 1

# Introduction

## 1.1 History

In recent years, use of robots in several industries is increasingly common. Robots are required to replace men in dangerous, boring or arduous tasks. Within these robots, UAVs can be found.

An UAV (unmanned aerial vehicle) is an unpiloted aircraft that can be operated remotely from a base station or fly autonomously based on a pre-programmed flight plan using a prearranged algorithm. This means that it does not require constant human input to maintain attitude and position, freeing the operator who can be focus on other tasks. It is not necessary for the operator to be a trained pilot, since human input does not directly involve any flight control data. Compared with manned vehicles, these aircraft are more maneuverable and operating costs may be lower. Moreover, they can avoid the risk inherent in manned flight in hostile environments in conditions of low visibility or in adverse weather conditions. However, accident rates in UAVs are still over 100 times than that of manned aircraft. Therefore, improved safety and reliability is still required.

UAVs are less expensive and simpler to build than airplanes. They are also more discreet and its lost is not as sensitive or expensive as a conventional vehicle. Their size can vary from a few centimeters to several meters and their weight from various grams to several tones. Their shape and type of propulsion are varied, for instance, some of them are equipped with reactors and some others with propellers or rotors.

Major advances in the field of unmanned aircraft, as in general aviation have arisen from wars between great powers. The first pilotless aircrafts were developed shortly after the First World War by the American Navy and Army and resembled modern cruise missiles. They were gasoline fueled propeller driven biplanes with a guide system composed of a gyroscope and a barometer/altimeter. UAVs were used for surveillance in the Vietnam War equipped with simple cameras and later with night photo and electronic intelligence. In 1982, the Israeli Air Force was able to destroy a big number of Syrian aircrafts with minimal losses by using UAVs that were capable of providing real-time surveillance or acting as electronic decoys or jammers. Since then, UAVs have played a very important role in military operations being able to realize more and more different missions.

UAVs are also known as drones. However, this term is associated with military applications reason why some manufacturers rebranding the technology in conjunction with their ground-

based control stations as unmanned aerial systems (UAS). They don't much like the term "unmanned" aerial vehicle either, because it implies that there is no pilot at all. They are sometimes called remotely piloted vehicles (RPVs), or remotely piloted aerial systems (RPAS).

Nowadays, major military and aerospace organizations and enterprises are developing innovative technologies to enhance UAV's control, communications technologies, sensors, propulsion, processors, data link systems or weaponry.

## 1.2 UAV applications

Due to its high versatility, UAVs have been envisaged for a variety of applications both as individual vehicles and in multiple vehicle teams. In the past, the use of UAVs has been mostly related to military applications, as it has been seen. Nowadays, the interest in UAVs in civil applications is growing as a result of the reduction in costs of the technology involved.

- Military missions:

A huge number of UAVs are used in the military field. Some of the greatest features when using UAVs in military missions are that there is no risk of endangering human lives and that they can carry out a wide range of applications at minimal cost. In espionage, border patrolling, coast guarding and security intelligence, target simulation (to confuse the enemy); UAVs are quite used, from the smallest ones (difficult to detect by radar) to the largest.

- Supervision:

One of the most valued skills of some types of UAVs is its maneuverability. This can be used for inspecting civil works such as large buildings, bridges, nuclear plants or other major structures. Vehicles are usually equipped with video cameras that allow to observe the structure from different angles without risking human lives. The execution time is also reduced since there is no need to build an external structure such as scaffolds to have access to a particular point.

UAVs equipped with thermographic cameras are also being used to monitor power lines. These devices serve the purpose of flying very close to the cables, which helicopters can't do safely and without risk to people. Security guards and policemen can use these systems to monitor a particular area or support any specific task. They can fight against illegal activities such as drug trafficking, illegal immigration, illegal fishing or recognizing chemical substances present in the ground.

- Aerial photography:

Aerial photography or aerial data acquisition is an important area because a lot of information can be extracted from the field. UAVs are used in environmental journalism or film making as they can obtain the same sharp videos and photos as a helicopter but much cheaper.

- Recognition of disasters:

Being unmanned vehicles, UAVs can be used for the initial recognition of disasters such as nuclear explosions, earthquakes, floods, volcanic eruptions or other situations where the integrity of the pilots could be in danger. They can also be a determining and effective agent in fire monitoring (before, during and after the fire) flying both in daytime and at night. They can be useful in locating people or delivering small objects in complex environments and difficult to access.

- Agriculture, forestall and environmental applications:

UAVs are used in crop monitoring which works to optimize the management of a plantation in what it is called "precision agriculture". They can help to better assess optimal planting density and to make decisions about the use of fertilizers and irrigation frequency. Besides, they can perform an early detection of diseases and pests. Instead of indiscriminately sprinkle insecticides from an airplane, drones, equipped with thermal cameras could detect which plants are sick and need treatment, as they give off less heat because of fungi infections.

Unmanned aircraft can also be prepared to take in-flight data about water quality in rivers and lakes, to monitor and observe wildlife in a given area or to control and protect fish stocks. In mines, for example, high-definition cameras drones can be used to create three-dimensional maps. Then, software is used to calculate how much material has been removed, allowing miners to adjust their production estimations.

Concerning environmental impact, they are useful to detect the emission of polluting gases, hydrocarbons and other toxic pollutants, as well as measure air quality.

In case it crash because of adverse environmental phenomenon, there would be only material damages and as they are electric, the possibility of fire would be minimal.

## 1.3 UAV in the present industry

Despite all these wide and as yet unlimited range of applications, many barriers specially technical and legislative must be overcome before drones can be used in civil airspace. Integration of UAS into non-segregated airspace will be a long term activity that involves many stakeholders in topics such as licensing and medical qualification of UAS crew, technologies for detect-and-avoid systems, frequency spectrum, separation standards form other aircraft etc. In addition, a climate of fear and uncertainty accompanies these vehicles. In the past few years, people have witnessed research into and development of drones capable not just of killing people at distance, but what are effectively flying with cameras, microphones, equipment capable of intercepting mobile phone data or with targeting and tracking capabilities. Besides this possible lack of privacy, the appearance of remote controlled aircraft raises a whole set of questions in case an accident or serious incident in civil operation occurs; behind the drones flying in the airspace must always be someone responsible.

### 1.3.1 Legislation in Europe

The European Commission has long subsidized research, development and international cooperation among drone manufacturers. A "roadmap" (Roadmap for the integration of civil RPAS into the European Aviation System, 2013) including a target date of 2028 for full integration of drones into commercial airspace was published in June 2013. This integration must be done ensuring a responsible, considered and effective use of the European civil airspace. The author of the roadmap was the European Remotely Piloted Aerial Systems Steering Group which includes industry representatives from Global Aerospace, Safran, Indra, EADS, Alenia etc. The EU is basing its own timeframe on that of the International Civil Aviation Organization, with which it will also work to develop global standards for drone flight.

Regarding the "light" UAS, a group of national authorities (JARUS) under the leadership of Netherlands and in cooperation with EASA, EUROCONTROL and SESAR JU are developing harmonized operational and technical regulations for less than 150 kg UAS.

### 1.3.2 Legislation in USA

In the United States, the leading country in the development of this technology, aviation authorities have not opened the airspace to domestic drones for commercial exploitation. The Federal Aviation Administration of the United States (FAA) asserts that rules to address the special safety challenges associated with unmanned aircraft need to be set before they can share the sky with manned aircraft. The agency has been working on those regulations for the past decade. However, legislation on commercial use in the country is not expected until after 2015. Manufacturers are looking forward to the regulation to launch into a market, which according to the International Association for Unmanned Vehicles Systems could move 82,000 million dollars in the next 10 years and up to 100,000 jobs.

Yet the FAA permits hobbyists to fly model aircrafts that have so improved in technology that they're little different from small drones. They are becoming so prevalent and affordable that the FAA has issued voluntary guidelines for hobbyists, including staying away from airports, flying no higher than 120 meters and staying within the line of sight of the operator to make sure they are not being used in a reckless way. They can also be used by government agencies. However, if non-government agencies want to operate an UAS for non-recreational purposes in the United States, they must obtain a Certificate of Authorization (COA) to operate in national airspace. At the moment, COAs require a public entity as a sponsor. (Federal Aviation Administration)

The FAA estimates that by the end of this decade there will be 30,000 drones flying the skies in that country, operated by the police, rescue teams, journalists, scientists, real estate agents and citizens. It is a very high number when it is consider that every day around 50,000 planes fly there. In war, the U.S. Air Force, already trains more drone operators than pilots.

### *1.3.3 Drones in the market*

In this section, some of the drones that different companies worldwide have developed are described. It starts with the large range military drones whose market is dominated by the USA and Israel. The European creations to be launched in this market are also reflected to end up with the smaller ones.

- General Atomics Aeronautical

The two most widely-used attack drones are the MQ-1 Predator and the upgraded MQ-9 Reaper (Predator B), both developed by the American military contractor General Atomics Aeronautical Systems. Predator and Reaper are prized for their ability to hover thousands of feet above a target for hours and relay high-resolution live surveillance. The Reaper has a range of 3,682 miles, an operational altitude of 50,000 ft, and a maximum flight time of 27 hours, making it especially useful for long-term operations with a maximum speed of about 276 miles per hour.


**Figure 1.3-1 Reaper**

- Northrop Grumman

It is an American global aerospace and defense technology company. In contrast to Predator, which is remotely piloted based on satellite navigation by pilots located miles away, the Global Hawk operates virtually autonomously. The user merely hits the button for “take off” and for “land”, while the UAV gets directions via GPS and reports back with a live feed.


**Figure 1.3-2 Global Hawk**

A variation of the Global Hawk, has been developed and built for the German Ministry of Defense in cooperation with Germany and EADS. The Euro Hawk has successfully completed its first tests flights but it is not yet certificate.

- Boeing

The American company Boeing is also part of the world of drones. Among others, it develops the Phantom Eye. It has proven the exceptional fuel economy of the liquid-hydrogen propulsion system. It is capable of flying at 20000 feet high for up to four days.

Figure 1.3-3 Phanton Eye

- Israel Aerospace Industries (IAI)

U.S. aerospace and defense manufacturing firms have a significant lead in military UAS, but Israel is also a strong competitor. The IAI has developed the Heron family. Its drone Eitan is the largest operating drone in flight, with endurance of 20 hours and up to 12 kilometers. It has the size of a Boeing 737, with wingspan 26 meters.

Figure 1.3-4 Eitan

- Cassidian (Airbus Group)

Cassidian is the European leader company in this high-tech segment. Barracuda is a multi-sensor system that offers an operating radius of up to 200 km, a maximum flight speed of Mach 0.6, and its test instrumentation can weigh up to 300 kg. Within the type MALE (Medium Altitude Long Endurance) drones, it is worth mentioning Harfang and Talarion.

Figure 1.3-5 Barracuda

The company also builds small UAV for civil purposes such as monitoring critical infrastructures and major events, reconnaissance, surveillance, search and rescue… The Copter4 is included in this group. It can operate at a range of 30 km with a 5 kg payload for more than 90 minutes at a time.

Figure 1.3-6 Copter4

- Dassault Aviation

nEUROn is the European unmanned combat air vehicle (UCAV) demonstrator for the development, integration and validation of UCAV technologies and is not for military operational deployment. The aircraft originally developed by the French Dassault, has the support of Italy (Alenia Aermacchi), Greece (HAI), Switzerland (RUAG), Sweden (Saab) and Spain (EADS CASA). The drone measures 10 meters long, has a wingspan of 12.5 meters and is powered by an engine manufactured by Rolls-Royce. It weighs about 500 kg unladen.

Figure 1.3-7 nEUROn

- BAE Systems

Taranis is a semi-autonomous unmanned warplane to fly intercontinental missions which is being designed by the British company BAE systems. It is one of the biggest and fastest UAV in existence.It will use stealth technology which makes it less visible to radar, infrared, sonar or other detection methods. The first flight tests have already taken place. However, the Taranis is planned to be operational after 2030 and it will be used along with manned vehicles.

Figure 1.3-8 Taranis

- Parrot SA

Parrot AR.Drone is a small radio controlled quadcopter with cameras attached to it, designed to be controllable by smartphones or tablet devices. In June 2013, Parrot announced they have sold over 500,000 AR.Drone quadcopters.

Figure 1.3-9 Parrot

- 3D Robotics

This American company has a catalog of several models ready to fly. IRIS, is a 'quadcopter' radio controlled that flies automatically according to an established flight plan. The route planning is surprisingly easy thanks to free software. On a real Google map the launch point is selected with the cursor as well as the different waypoints that it must reach.

Figure 1.3-10 Iris

- Others

The market is so wide that is impossible to consider all. It is even broader for small drones that are being developed in many firms and research centers eager to join the world of drones. In Japan, for example, the Yamaha Motor Company's RMAX helicopter drones have been spraying crops for 20 years. The company Deutsche Post DHL is testing a "Paketkopter" drone that could be used to deliver small, urgently needed goods in hard-to-reach places. AeroVironment has developed a tiny drone ordered by the Pentagon, the Nano Hummingbird. It weighs 19 grams and can fly more than 27 kilometers / hour for eight minutes. Facebook is in talks to buy Titan Aerospace, a maker of solar-powered drone-like satellites, to step up its efforts to provide Internet access to remote parts of the world. As the last ones, many other manufactures could follow.

# Chapter 2

# Quadrotor System

Quadcopters are one of the UAVs that are major focuses of active researches in recent years due to its small size, light weight, effortless assembling of mechanical structures and their ability to complete tasks efficiently and autonomously. The quadrotor has four equally-spaced rotors in cross configuration. Each propeller is driven by an electric motor, which is powered by an onboard battery.

However, controlling four independent rotors is very difficult without electronic assistance. Today, advanced electronics have helped to solve this limitation. These aircrafts can be autonomous or remotely operated. A computer or microcontroller on board and a variety of accurate sensors including accelerometers, gyroscopes and magnetometers are responsible for maintaining stability of the quadcopter in the air. The quadrotor has six degrees of freedom (three translational and three rotational) and only four independent inputs which are the rotor's speeds. Therefore, it is an underactuated and dynamically unstable system.

## 2.1 Basic concepts

This section will discuss, at first, the advantages and disadvantages of the currently Quadcopters. Then, the quadcopter dynamics will be explained. Adjusting separately the rpm of each rotor, the vehicle will move in one direction or another. Finally, some assumptions that will be present during the project are defined.

### *2.1.1 Advantages and drawbacks*

Its ability to Vertically Take-Off and Land (VTOL) and hovering categorizes the quadcopter as a helicopter UAVs. However, quadcopters have some advantages over conventional helicopters like simpler mechanical design. They change direction by manipulating the individual propeller's speed and do not require cyclic and collective pitch control.

One characteristic to note is the high maneuverability of the vehicle, which makes them suitable to hover, take off, fly and land indoors or in limited spaces. By enclosing the rotors within a frame, rotors can be protected from breaking during collisions, reducing the

possibilities of damaging the vehicle, its operators, or its surroundings. Due to its capacity of vertical flight and hovering, the quadcopter can also be chosen when data acquisition is needed.

Unfortunately, the advantage of being maneuverable makes controls very complex and difficult, requiring sophisticated sensors and fast on-board computation. A quadrotor is unstable and impossible to fly in full open loop system. Therefore, a control model that provides a good stability in autonomous flight, or at least to assist the pilotage of the vehicle, providing high maneuverability and robustness against external disturbance, is required.

Its load capacity is quite high compared to the weight of the entire system. The thrust generated by a rotor, and therefore the weight that can be lifted, rise as its diameter increases. However, there is a limit imposed by the compressibility effects. Thanks to its four rotors, the quadcopter can lift more weight without the need to augment the diameter excessively. Nevertheless, it is obvious that having four rotors instead of one or two will increase the takeoff weight. As weight is proportional to the hover ability, higher levels of power consumption will be needed to hover.

Another advantage in quadrotors is that gyroscopic effects are very small. As every rotating body, the rotor of a helicopter is affected by gyroscopic effects. These effects depend on the rpm and the direction of rotation. In the case of the quadcopter, two rotors are turning clockwise and the other two counterclockwise. This way, when rpm are identical in the four rotors, as in the hovering, gyroscopic effects are cancelled.

The fact that it has four small propellers reduces the torque on the system which means that the blades can be driven at higher velocities without producing additional mechanical vibrations. This results in an increase of the motor efficiency and puts less stress on the mechanical components of the craft.

The autonomy of the system is an important characteristic that must be taken into account. However, quadrotor's autonomy is not very good. A consequence of having more rotors is the bigger power consumption. To increase autonomy, bigger batteries would be needed which leads to higher take off weights. The reduced flight endurance is an important limitation of the quadcopter, batteries need to provide more capacity and be smaller to solve this problem.

### *2.1.2 Principles of quadcopter's flight*

As it has already been mentioned, the aircraft has four rotors to sustain itself in air. Its rotation and position is very significant. Motor one and three are placed along the X-axis and rotate the in counterclockwise direction. Motor two and four are placed on the Y-axis and rotate in clockwise direction, as it can be observed in the following picture:

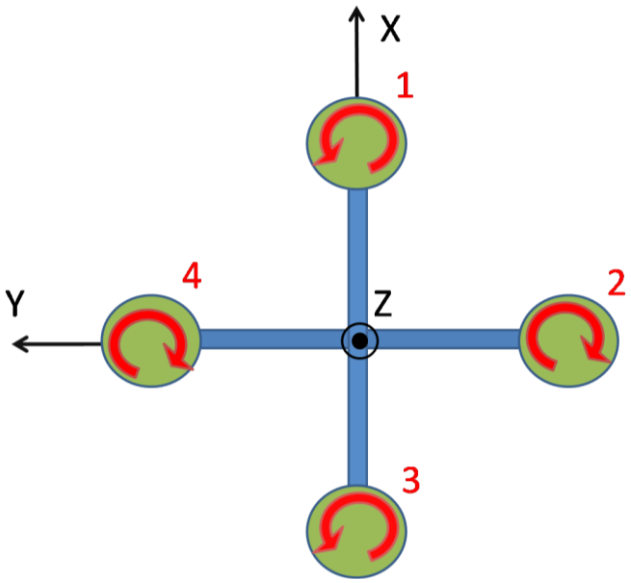



Figure 2.1-1: Quadcopter's body axis

The fixed body frame is represented in black and the red arrows indicate the direction of rotation of the propellers. The central vertical axis is parallel to the rotating axes of the rotors. This characteristic is going to greatly simplify the geometry and reduce the complexity of the equations of motion. They have fixed-pitch blades and the air flows downwards (to get an upward lift). The vehicle tilts towards the direction of the slow spinning rotor, which enables acceleration along that direction.

- ALTITUDE CONTROL

From a hovering condition, the ascent or descent of the quadcopter is achieved by increasing (or decreasing) rotational speed of each of the four rotors in the same proportion. This will increase (or decrease) the total thrust applied to the aircraft allowing ascent (or descent). To ascend, the lift force generated by the 4-propeller torques must be greater than the weight force generated by the gravitational attraction.

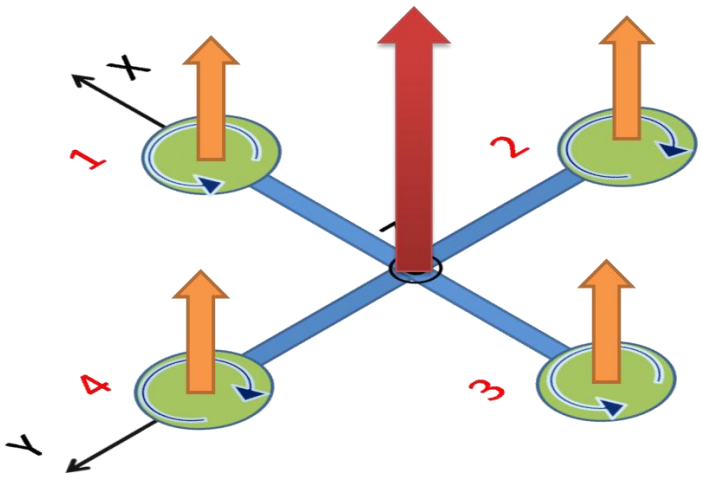



Figure 2.1-2: Altitude control

- ATTITUDE CONTROL

Movement in the horizontal plane is done by tilting the quadrotor in the desired direction. When the quadrotor is tilted the lift force is no longer aligned with the earth frame z-axis. This force can be decomposed into a z-axis component and a horizontal component. The last one generates acceleration in the horizontal plane.

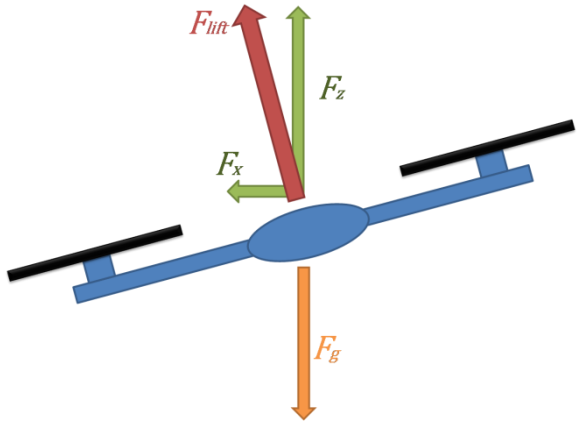



Figure 2.1-3: Lift force decomposed into a z-axis component and a horizontal component

The three attitude angles (roll, pitch, yaw) of the quadrotor are controlled by changing the velocity of different pairs of motors.

Roll control

It is produced when the vehicle is moving left or right. The rotation about the X axes, roll, is achieved by increasing the thrust of the rotor 4 and reducing the thrust of the rotor 2 to obtain a positive roll. The negative roll is achieved by increasing the thrust of the rotor 2 and reducing the thrust of the rotor 4. In the following picture, the negative roll is represented in red. The negative torque in x axis generates a movement along the y positive axis. In the picture the size of the orange arrows is proportional to the thrust generated by each rotor.

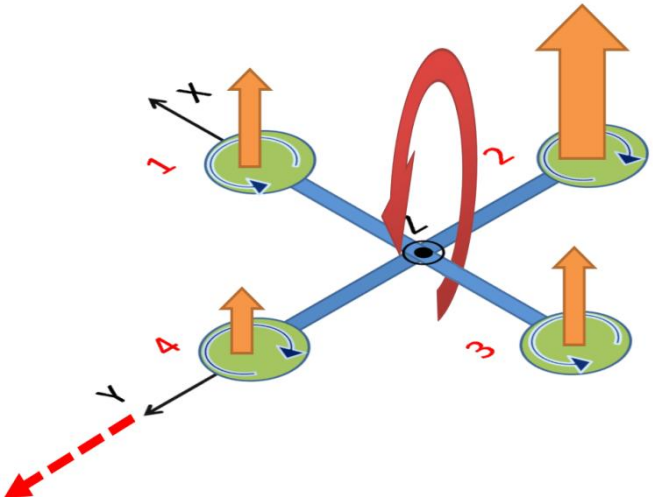



Figure 2.1-4: Roll control

Pitch control

It is the movement which allows the forward motion and backward. Rotation about the Y axis or pitch is obtained similarly. The positive (negative) pitch is achieved by increasing (decreasing) the thrust of the rotor 3 and reducing (increasing) the thrust of the rotor 1. In the picture, a positive torque about the y axis is applied, which generates a movement along the positive x axis.

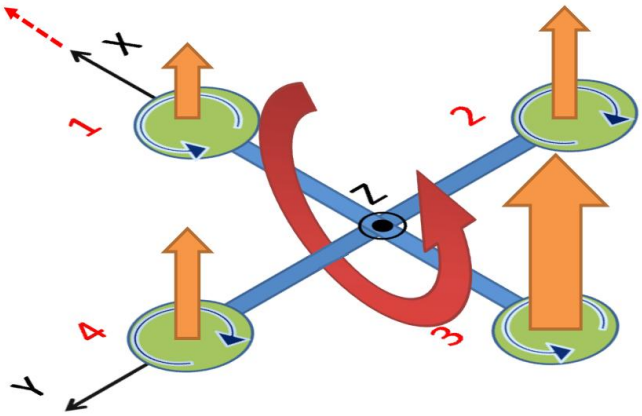



Figure 2.1-5: Pitch control

Yaw control

The yaw motion refers to the movement of the vehicle when it rotates on its vertical axis. The quadrotor achieves this movement increasing (or decreasing) power rotation equally in rotors 1 and 3 and decreasing (or increasing) in equal magnitude engines 2 and 4. The increased drag of the motors at higher speed (higher thrust) creates a reaction moment that rotates the body about the z axis. The rotation is then opposite to the rotation of the propellers at higher speed.

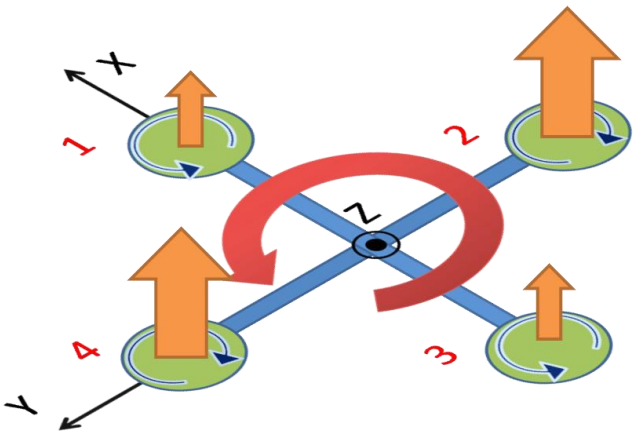



Figure 2.1-6: Yaw control

### 2.1.3 Assumptions of the model

The quadcopter has a complex physical configuration. This means that without simplifying some assumptions, it would be very difficult to obtain its mathematical model. The assumptions that are used to obtain the model are the following:

- The structure is supposed to be rigid, as well as the propellers. The truth is that the blades are flexible, so the thrust is not parallel to the axis of the rotor.
- The quadrotor's structure is perfectly symmetric, so the matrix of inertia will be diagonal.
- The center of mass and the body fixed frame origin are assumed to coincide.
- Thrust and drag are proportional to the squared of the propeller speed. Actually, they are heavily dependent with the airflow through the rotor.
- Ground effect is neglected.

## 2.2 Equations of motion

To be able to simulate a system, it is necessary to create a mathematical model that represents the system dynamics in a simplified way. At the end of this section, a mathematical model for an under actuated six-degree of freedom quadrotor will be obtained. These equations will form the Simulink model of the system that will be represented in section 2.3.

The equations that described the quadcopter dynamics vary depending on whether the center of mass is displaced by a given position from the origin of rotation body-fixed frame (Raffo, 2011) , or whether rotor dynamics (BOUABDAHALL, 2007), (Yanmin Chen, Yongling He and Minfeng Zhou, 2013) or aerodynamic forces (G. V. Raffo, M. G. Ortega, and F. R. Rubio., 2010c), (Hoffmann) are included. However, when simulating the system, most cases take into account the assumptions shown in the previous section. This way, a simple model that enables the simulation of the flight controls in Simulink is obtained.

The model will be derived using two methods: Newton-Euler formalism and the Lagrange formalism. It consists of two main parts, the first one is the rotational system of equations and the second is the translational system of equations. Before going directly to the equations, the reference systems that will be used are defined and the angular body rates are obtained.

### 2.2.1 Quadrotor Coordinate Frames

To describe the position and orientation of the quadcopter, various reference frames are needed. For example, the aerodynamics forces and torques are given in the body frame, accelerometers and gyroscopes also measure the information with respect to the body frame, but the GPS measures of position or ground speed are given with respect to the inertial frame.

To transform one coordinate frame into another, different rotations and translations are done (Mecánica de vuelo, 2012). A valid assumption for quadrotors is to use a flat and non-rotating earth. The coordinate frames that will be used are:

- Inertial frame

It is an earth fixed coordinate system with origin at a define location. The unit vector $\mathbf{i}^i$ is directed to the North, the $\mathbf{j}^i$ to the East and the $\mathbf{k}^i$ is directed upward forming a clockwise trihedral with the orders.

- Vehicle frame A

The origin is fixed in the centre of mass of the quadcopter. However, the axes of this system are aligned with the inertial frame: $\mathbf{i}^A$ is also directed to the North, the $\mathbf{j}^A$ to the East and the $\mathbf{k}^A$ is directed upward forming a clockwise trihedral with the orders.

- Vehicle frame 1

The origin is the same as in frame A. However, the system is rotated positively around $\mathbf{k}^A$ the yaw angle $\psi$. The rotation matrix that allows passing from A system to system 1 is:

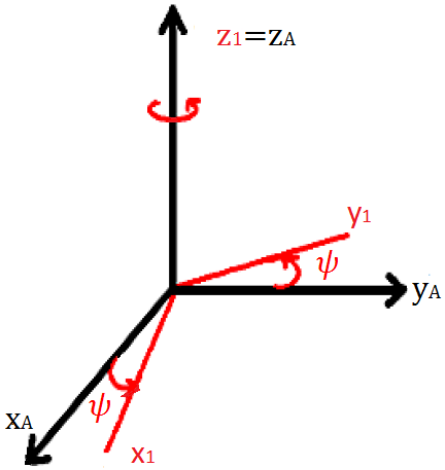



Figure 2.2-1 Psi rotation

$$\vec{L}_1 = R_{1A} \cdot \vec{L}_A$$

$$\begin{bmatrix} \vec{\imath}_1 \\ \vec{\jmath}_1 \\ \vec{k}_1 \end{bmatrix} = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath}_A \\ \vec{\jmath}_A \\ \vec{k}_A \end{bmatrix} \quad (1)$$

- Vehicle frame 2

The origin of the vehicle frame 2 is again the centre of gravity. The system is obtained by rotating right-handed the vehicle frame 1 around $\mathbf{j}_1$ the angle θ. The rotation matrix that allows passing from system 1 to system 2 is:

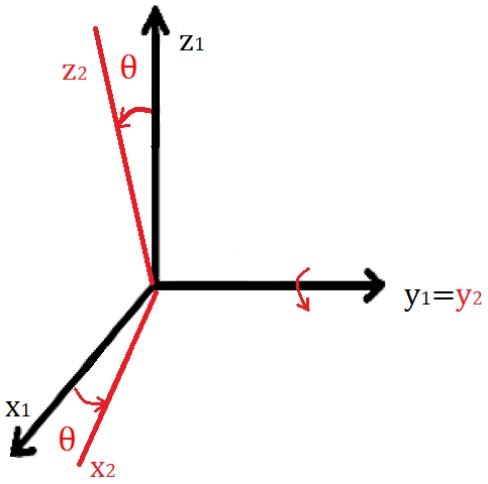



Figure 2.2-2: Theta rotation

$$\vec{L}_2 = R_{21} \cdot \vec{L}_1$$
$$\begin{bmatrix} \vec{i}_2 \\ \vec{j}_2 \\ \vec{k}_2 \end{bmatrix} = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \cdot \begin{bmatrix} \vec{i}_1 \\ \vec{j}_1 \\ \vec{k}_1 \end{bmatrix} \quad (2)$$

- Body frame

The origin continues being the same. To obtain the body frame, the previous system is rotated right-handed around $\mathbf{i_2}$ the angle ϕ. In this system, $\mathbf{i_b}$ points out to the nose of the quadcopter (rotor 1), $\mathbf{j_b}$ points to the rotor 4 and $\mathbf{k_b}$ points upwards forming a trihedral with the others. The transformation from 1 to the body frame is given by:

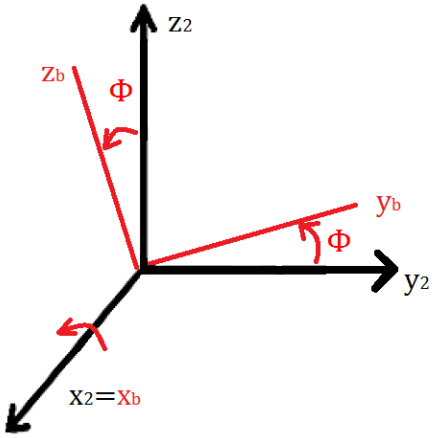



Figure 2.2-3: Phi rotation

$$\vec{L}_b = R_{b2} \cdot \vec{L}_2 \quad (3)$$

$$\begin{bmatrix} \vec{\imath}_b \\ \vec{\jmath}_b \\ \vec{k}_b \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath}_2 \\ \vec{\jmath}_2 \\ \vec{k}_2 \end{bmatrix}$$

To obtain the global rotation matrix that converts the vehicle frame A to the body frame, the individual rotation matrixes are multiple in a particular order with the following result:

$$\vec{L}_b = R_{b2} \cdot R_{21} \cdot R_{1A} \cdot \vec{L}_A = R_{bA} \cdot \vec{L}_A$$

$$R_{bA} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \cdot \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_{bA} = \begin{bmatrix} cos\theta cos\psi & cos\theta sin\psi & -sin\theta \\ sin\phi sin\theta cos\psi - sin\psi cos\phi & sin\phi sin\theta sin\psi + cos\phi cos\psi & sin\phi cos\theta \\ sin\theta cos\phi cos\psi + sin\phi sin\psi & sin\theta cos\phi sin\psi - sin\phi cos\psi & cos\phi cos\theta \end{bmatrix} \quad (4)$$

The transformation matrixes between to reference systems are orthogonal which means that the inverted matrix is equal to the transposed matrix. This property makes very easy to pass from the body frame to the vehicle frame A by using a new rotation matrix. It will be useful as the vehicle frame A has the same orientation as the inertial frame, as it has been seen.

$$R_{Ab} = \begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - sin\psi cos\phi & sin\theta cos\phi cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & sin\theta cos\phi sin\psi - sin\phi cos\psi \\ -sin\theta & sin\phi cos\theta & cos\phi cos\theta \end{bmatrix} \quad (5)$$

### *2.2.2 Quadrotor's angular body rates*

The absolute angles $\phi$, $\theta$ and $\psi$ are defined in different coordinate systems: 2, 1 and A respectively. Time variation of these angles is related with the angular body rates. These angles are p, which is the roll rate measured along $\mathbf{i_b}$; q is the pitch rate measured along $\mathbf{j_b}$; and r is the yaw rate measured along $\mathbf{k_b}$.

The conversion from one system to another is done as follows:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \dot{\phi}\vec{\imath}_2 + \dot{\theta}\vec{\jmath}_1 + \dot{\psi}\vec{k}_A \quad (6)$$

Using the rotation matrixes that were calculated in the previous section, vectors $\vec{\imath}_2$, $\vec{\jmath}_1$ and $\vec{k}_A$ can be expressed in body axes.

$$\vec{L}_2 = R_{2b} \cdot \vec{L}_b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi & cos\phi \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath}_b \\ \vec{\jmath}_b \\ \vec{k}_b \end{bmatrix}$$

$$\vec{\imath}_2 = \vec{\imath}_b \tag{7}$$

$$\vec{L}_1 = R_{12} \cdot R_{2b} \cdot \vec{L}_b = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath}_b \\ \vec{\jmath}_b \\ \vec{k}_b \end{bmatrix}$$

$$\vec{\jmath}_1 = cos\phi\, \vec{\jmath}_b - sin\phi\, \vec{k}_b \tag{8}$$

$$\vec{L}_A = R_{A1}.\, R_{12} \cdot R_{2b} \cdot \vec{L}_b = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \cdot \begin{bmatrix} \vec{\imath}_b \\ \vec{\jmath}_b \\ \vec{k}_b \end{bmatrix}$$

$$\vec{k}_A = -sin\theta\, \vec{\imath}_b + sin\phi cos\theta\, \vec{\jmath}_b + cos\phi cos\theta\, \vec{k}_b \tag{9}$$

The equations (7), (8) and (9) are introduced in equation (6) with the following results:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -sin\theta \\ 0 & cos\phi & sin\phi cos\,\theta \\ 0 & -sin\phi & cos\phi cos\theta \end{bmatrix} \cdot \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{10}$$

### *2.2.3 Forces and torques*

Forces and torques in the quadcopter are due to gravity and the four motors. There are not aerodynamic lifting surfaces so it will be assumed that aerodynamic forces are negligible.

- Motor's forces and torques

Each motor produces a force and a torque. The global thrust is the sum of the force produced by the four motors. This value is going to be called $U_1$:

$$U_1 = F_1 + F_2 + F_3 + F_4 \tag{11}$$

Roll and pitch torques produced by the propeller's forces are written in the equations (12) and (13) where *l* is the distance between the z axis and the propeller's axis.

$$\tau_\phi = l(F_4 - F_2) \tag{12}$$

$$\tau_\theta = l(F_3 - F_1) \tag{13}$$

The other torque is given by the third Newton's Law. The drag of the propeller produces a yawing torque on the quadcopter's body. The direction of this torque will be in the opposite direction of the motion of the propeller.

$$\tau_{\psi} = -\tau_1 + \tau_2 - \tau_3 + \tau_4 \tag{14}$$

The forces and torques are equal to a coefficient multiplied by the propeller speed squared in radians per second. The coefficients *b* and *d* will be calculated in section 2.4.3. If these values are introduced in the equations (11), (12), (13), (14) and the roll, pitch and yaw torques are called $U_2$, $U_3$ and $U_4$, the equations become:

$$\begin{aligned} U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= l \cdot b(\Omega_4^2 - \Omega_2^2) \\ U_3 &= l \cdot b(\Omega_3^2 - \Omega_1^2) \\ U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{aligned} \tag{15}$$

Furthermore, the motor's rotation generates a gyroscopic torque that must be taken into account. Each motor can be considered as a hard disk rotating about its Z axis with a velocity $\Omega_i$. The motor's rotation axis moves at the reference axis' angular velocity, which produces the following gyroscopic movements:

$$\tau = -J_r \cdot \left(\omega \times \vec{k}\right) \cdot (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) \tag{16}$$

where $J_r$ is the inertia moment of the motor around its rotation axis and $\omega$ is the angular velocity of the helicopter, expressed in the body frame.

Equation (16) has the following result:

$$\begin{aligned} \tau_{\phi} &= J_r \cdot \dot{\theta} \cdot (\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4) \\ \tau_{\theta} &= J_r \cdot \dot{\phi} \cdot (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) \end{aligned} \tag{17}$$

- Gravitational force

Gravity also exerts a force on the COG of the quadcopter. This force is expressed in inertial frame as:

$${f_g}^i = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} \tag{18}$$

with g being the gravitational constant. This force can be converted to body frame using the rotation matrix of the equation (4).

$${f_g}^b = R_{bA} \cdot \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} = \begin{pmatrix} mgsin\theta \\ -mgsin\phi cos\theta \\ -mgcos\phi cos\theta \end{pmatrix} \tag{19}$$

### 2.2.4 Quadrotor dynamics: Classical Mechanics method

In this section, the Newton-Euler formulation will be used to obtain the dynamic equations that describe the relationship between force and motion.

- Momentum's conservation

Newton`s second law states that the net force on an object is equal to the rate of change of its linear momentum in an inertial reference frame. In this case, mass is constant so that the equation can be express as:

$$\vec{F} = m \cdot \frac{d\vec{v}}{dt_i} \tag{20}$$

The vector $\vec{v}$ has the components u, v and w, which are the body frame velocities measured along $\mathbf{i_b}$, $\mathbf{j_b}$ and $\mathbf{k_b}$ respectively. The derivation in the equation is done in inertial frame, in order to solve it, the Coriolis equation is used. In the equation, $\vec{\omega}_{bi} = (p, q, r)$ is the angular velocity of the body frame with respect to the inertial frame.

$$\vec{F} = m \cdot \left( \frac{d\vec{v}}{dt_b} + \vec{\omega}_{bi} \times \vec{v} \right) \tag{21}$$

Replacing terms in the above equation, it is obtained:

$$\begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = m \cdot \begin{pmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{pmatrix} \tag{22}$$

Regrouping the equation:

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \frac{1}{m} \cdot \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \tag{23}$$

- Angular momentum

This theorem states that the applied torque ($\vec{G}$) is equal to:

$$\vec{G} = \frac{d\vec{h}}{dt_i} = \frac{d(\boldsymbol{I}\vec{\omega})}{dt_i} \tag{24}$$

Using Coriolis again:

$$\vec{G} = \frac{d\vec{h}}{dt_b} + \vec{\omega}_{bi} \times \vec{h} \tag{25}$$

**I** is the constant inertia matrix which is given by:

$$I = \begin{pmatrix} \int (y^2 + z^2)\, dm & -\int xy\, dm & -\int xz\, dm \\ -\int xy\, dm & \int (x^2 + z^2) dm & -\int yz \quad dm \\ -\int xz\, dm & -\int yz\, dm & \int (x^2 + y^2) dm \end{pmatrix} = \begin{pmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{xy} & I_y & -I_{yz} \\ -I_{xz} & -I_{yz} & I_z \end{pmatrix} \quad (26)$$

The quadcopter is symmetric in the three axes, therefore $I_{xy}, I_{yz}$ and $I_{xz}$ are equal to cero. Besides, the tensor I is constant because mass doesn`t change. If in equation (25), the term $\frac{d\vec{w}}{dt_b}$ is isolated; the result is:

$$\frac{d\vec{\omega}}{dt_b} = I^{-1} \cdot \left(\vec{G} - \vec{\omega}_{bi} \times \vec{h}\right) \quad (27)$$

If all the values are introduced, where the components of the torque are $\vec{G} = (\tau_\phi, \tau_\theta, \tau_\psi)$ it is obtained:

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} 1/I_x & 0 & 0 \\ 0 & 1/I_y & 0 \\ 0 & 0 & 1/I_z \end{pmatrix} \cdot \left[ \begin{pmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} - \begin{pmatrix} qI_z r - rI_y q \\ rI_x p - pI_z r \\ pI_y q - qI_x r \end{pmatrix} \right] \quad (28)$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{I_y - I_z}{I_x} qr \\ \frac{I_z - I_x}{I_y} rp \\ \frac{I_x - I_y}{I_z} pq \end{pmatrix} + \begin{pmatrix} \tau_\phi / I_x \\ \tau_\theta / I_y \\ \tau_\psi / I_z \end{pmatrix} \quad (29)$$

The angles $\phi$ and $\theta$ are supposed to be very little. If these angles are neglected in the equation (10), angular body rates can be taken as equal to the time variation of $\phi, \theta$ and $\psi$ and the equation (29) can be written as:

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \frac{I_y - I_z}{I_x} \dot{\theta}\dot{\psi} \\ \frac{I_z - I_x}{I_y} \dot{\psi}\dot{\phi} \\ \frac{I_x - I_y}{I_z} \dot{\phi}\dot{\theta} \end{pmatrix} + \begin{pmatrix} \tau_\phi / I_x \\ \tau_\theta / I_y \\ \tau_\psi / I_z \end{pmatrix} \quad (30)$$

If the gyroscopic torques (17) as well as the values U2, U3 and U4 (15) are introduced in the equation (30), the attitude equations that will be introduced in the quadcopter's model are obtained.

$$\ddot{\phi} = \frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi} + \frac{J_r\dot{\theta}}{I_x}\cdot(\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4) + \frac{lb}{I_x}(\Omega_4^2 - \Omega_2^2)$$

$$\ddot{\theta} = \frac{I_y - I_z}{I_y}\dot{\theta}\dot{\psi} + \frac{J_r\dot{\phi}}{I_y}\cdot(-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) + \frac{lb}{I_y}(\Omega_3^2 - \Omega_1^2) \quad (31)$$

$$\ddot{\psi} = \frac{I_x - I_y}{I_z}\dot{\phi}\dot{\theta} + \frac{d}{I_z}(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2)$$

Some works as (BOUABDAHALL, 2007) also include in the attitude equations, the rotor dynamics; not only the gyroscopic effects, as in this work, but also terms involved in the next formulas:

$$L\frac{di}{dt} = u - R_{mot}\,i - K_e\,\omega_m$$

$$J_m\frac{d\omega_m}{dt} = \tau_e - \tau_f$$

It includes terms as the voltage ($u$), the motor resistance ($R_{mot}$), electric constant ($K_e$), current intensity ($i$) or friction and electromagnetic torques ($\tau_f$ and $\tau_e$).

The rotor dynamics has not been introduced in the model for two principal reasons. The first one is that the constants needed to reproduce the rotor dynamics are not available and complex experiments would have had to be performed to get them, which is beyond the aims of this work. Secondly, it should not be forgotten that the speed of the rotors depends strongly on the dynamics of the motor(s) driving them. Any motor or engine has certain inertia to changes in its regime: the larger inertia, the larger time lag. Depending on the type of engine and its size, the time lag may differ in several orders of magnitude. Usually this is not a problem with electrical motors as the ones employed in this type of quadcopters that are small with a very little time lag. Inertia is therefore very little too, and the response is considered to be almost instantaneous. Large time lag is the reason why to control other types of quadcopters, as the ones powered by internal combustion, the rpm control is not the best method.

### 2.2.5 *Quadrotor dynamics: Lagrange method*

These formulas can also be obtained by using the Lagrangian Mechanics. It is a reformulation of classical mechanics introduced by Joseph Louis Lagrange in 1788. In Lagrangian Mechanics, the trajectory of an object is obtained by finding the trajectory that minimized the action. This path is the Lagrangian over time, which is the subtraction between the kinetic energy and the potential energy.

$$L = T - V = Total\ kinetic\ energy - Total\ potential\ energy \quad (32)$$

In its most general form, the Lagrange equations take the form:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \left(\frac{\partial L}{\partial q_j}\right) = 0 \quad (33)$$

Given an N particles system and a set of position vectors for the particles, it can always be found a number of generalized coordinates $q_j$ that better describes the energetic evolution of the system. There is one Lagrange equation for each generalized coordinate $q_j$.

With the equations above, all kinds of problems for conservative dynamical systems can be approached. For general systems in which there are both conservative and non-conservative forces it must be noticed that all forces present in the system cannot be expressed from a potential. In these systems, a new term that represents the generalized forces provided by non-conservative forces needs to be introduced.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \left(\frac{\partial L}{\partial q_j}\right) = \Gamma_i \quad (34)$$

Although mathematics required for Lagrange's equations might appear significantly more complicated than Newton's laws, in practice it's often easier to solve a problem using Lagrange equations. Some reasons are:

- It dispenses with the forces acting on different parts of the system
- The minimum generalized coordinates $q_j$ can be chosen as appropriate to take advantage of the symmetries of the system
- Constraint forces are incorporated into the geometry of the problem
- The Lagrangian function is a scalar, not a vector, from which the differential equations of motion are obtained

The generalized coordinates chosen for the quadcopter are the attitude angles phi (ϕ), theta (θ) and psi ($\psi$). The Lagrange's equations can be expressed as:

$$\Gamma_\phi = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\phi}}\right) - \left(\frac{\partial L}{\partial \phi}\right) = \tau_x \quad (35)$$

$$\Gamma_\theta = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \left(\frac{\partial L}{\partial \theta}\right) = \tau_y \quad (36)$$

$$\Gamma_\psi = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) - \left(\frac{\partial L}{\partial \psi}\right) = \tau_z \quad (37)$$

Firstly, the position in body frame, which is going to be represented by *x*, *y*, and *z*, is rotated to be expressed in earth axis. For that, the matrix $R_{Ab}$ expressed in equation (5) is used, obtaining the following results:

$$r_x = c\theta c\psi \cdot x + (s\phi s\theta c\psi - s\psi c\phi) \cdot y + (s\theta c\phi c\psi + s\phi s\psi) \cdot z$$

$$r_y = c\theta s\psi \cdot x + (s\phi s\theta s\psi + c\psi c\phi) \cdot y + (s\theta c\phi s\psi - s\phi c\psi) \cdot z \quad (38)$$

$$r_z = -s\theta \cdot x + s\phi c\theta \cdot y + c\theta c\phi \cdot z$$

These formulas are derived with respect to time:

$$\begin{aligned} v_x = {} & (-\dot{\theta}s\theta c\psi - \dot{\psi}c\theta s\psi) \cdot x \\ & + \left(-\dot{\psi}c\phi c\psi + \dot{\phi}s\phi s\psi - \dot{\psi}s\phi s\psi s\theta + \dot{\phi}c\phi c\psi s\theta + \dot{\theta}s\phi c\psi c\theta\right) \cdot y \\ & + (\dot{\psi}s\phi c\psi + \dot{\phi}c\phi s\psi - \dot{\psi}c\phi s\psi s\theta - \dot{\phi}s\phi c\psi s\theta + \dot{\theta}c\phi c\psi c\theta) \cdot z \end{aligned}$$

$$\begin{aligned} v_y = {} & (-\dot{\theta}s\theta s\psi + \dot{\psi}c\theta c\psi) \cdot x \\ & + \left(-\dot{\psi}c\phi s\psi - \dot{\phi}s\phi c\psi + \dot{\psi}s\phi c\psi s\theta + \dot{\phi}c\phi s\psi s\theta + \dot{\theta}s\phi s\psi c\theta\right) \cdot y \\ & + (\dot{\psi}s\phi s\psi - \dot{\phi}c\phi c\psi + \dot{\psi}c\phi c\psi s\theta - \dot{\phi}s\phi s\psi s\theta + \dot{\theta}c\phi s\psi c\theta) \cdot z \end{aligned} \quad (39)$$

$$v_z = -\dot{\theta}c\theta \cdot x + (\dot{\phi}c\phi c\theta - \dot{\theta}s\phi s\theta) \cdot y + (-\dot{\theta}c\phi s\theta - \dot{\phi}s\phi c\theta) \cdot z$$

For each one of the axis, the above formulas can be written as:

$$v_x = \begin{pmatrix} v_{x_x} & v_{x_y} & v_{x_z} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}; \quad v_y = \begin{pmatrix} v_{y_x} & v_{y_y} & v_{y_z} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix};$$
$$v_z = \begin{pmatrix} v_{z_x} & v_{z_y} & v_{z_z} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (40)$$

The square root of the magnitude of the velocity is given by:

$$v^2 = {v_x}^2 + {v_y}^2 + {v_z}^2 \quad (41)$$

$$\begin{aligned} v^2 = {} & \left({v_{x_x}}^2 + {v_{x_y}}^2 + {v_{x_z}}^2\right) \cdot x^2 + \left({v_{y_x}}^2 + {v_{y_y}}^2 + {v_{y_z}}^2\right) \cdot y^2 \\ & + \left({v_{z_x}}^2 + {v_{z_y}}^2 + {v_{z_z}}^2\right) \cdot z^2 + 2xy \cdot \left(v_{x_x}v_{x_y} + v_{y_x}v_{y_y} + v_{z_x}v_{z_y}\right) \\ & + 2xz \cdot \left(v_{x_x}v_{x_z} + v_{y_x}v_{y_z} + v_{z_x}v_{z_z}\right) + 2yz \cdot \left(v_{x_y}v_{x_z} + v_{y_y}v_{y_z} + v_{z_y}v_{z_z}\right) \end{aligned} \quad (42)$$

The total kinetic energy is given by:

$$T = \frac{1}{2}\int v^2 \, dm \quad (43)$$

The values of equation (39) are introduced in $v^2$. Moreover, the variables are regrouped in order to make visible the elements of the inertia matrix, obtaining:

$$
\begin{aligned}
T = {} & \frac{1}{2}\underbrace{\int (y^2+z^2)\ dm}_{I_x}\left[\dot{\psi}^2 s^2\theta - 2\dot{\psi}\dot{\phi}s\theta + \dot{\phi}^2\right] \\
& + \frac{1}{2}\underbrace{\int (x^2+z^2)\ dm}_{I_y}\left[\dot{\psi}^2 s^2\phi c^2\theta + 2\dot{\psi}\dot{\theta}s\phi c\phi c\theta + \dot{\theta}^2 c^2\phi\right] \\
& + \frac{1}{2}\underbrace{\int (x^2+y^2)\ dm}_{I_z}\left[\dot{\psi}^2 c^2\phi c^2\theta - 2\dot{\psi}\dot{\theta}s\phi c\phi c\theta + \dot{\theta}^2 s^2\phi\right] \\
& + \frac{1}{2}\underbrace{\int xy\ dm}_{\sim 0}\left[\dot{\psi}^2 s\phi s\theta c\theta + \dot{\psi}\left(\dot{\theta}c\phi s\theta - \dot{\phi}s\phi c\theta\right) - \dot{\theta}\dot{\phi}c\phi\right] \\
& + \frac{1}{2}\underbrace{\int xz\ dm}_{\sim 0}\left[\dot{\psi}^2 c\phi s\theta c\theta + \dot{\psi}\left(-\dot{\phi}c\phi c\theta - \dot{\theta}s\phi s\theta\right) + \dot{\theta}\dot{\phi}s\phi\right] \\
& + \frac{1}{2}\underbrace{\int yz\ dm}_{\sim 0}\left[-\dot{\psi}^2 s\phi c\phi c^2\theta + \dot{\psi}\left(\dot{\theta}s^2\phi c\theta - \dot{\theta}c^2\phi c\theta\right) + \dot{\theta}^2 s\phi c\phi\right]
\end{aligned}
\tag{44}
$$

As it can be seen, the terms of the inertia matrix (26) appear directly in the equations. The symmetry of the model allows neglecting the inertia products since the inertia matrix is considered diagonal. Besides, some algebraic identities are introduced to simplify the expression.

$$
T = \frac{1}{2}I_x\left(\dot{\phi} - \dot{\psi}sin\theta\right)^2 + \frac{1}{2}I_y\left(\dot{\psi}sin\phi cos\theta + \dot{\theta}cos\phi\right)^2 + \frac{1}{2}I_z\left(\dot{\psi}cos\phi cos\theta - \dot{\theta}sin\phi\right)^2 \tag{45}
$$

As for the total potential, it is calculated using the rotation matrix again obtaining the following result:

$$
V = g \cdot \int r_z dm \tag{46}
$$

$$
V = \int xdm(gsin\theta) + \int ydm(-gsin\phi cos\theta) + \int zdm(-gcos\phi cos\theta) \tag{47}
$$

Calculations of the Lagrange equation for the roll angle are described below, for the rest of the parameters the procedure is the same:

$$
\begin{aligned}
\frac{\partial T}{\partial \phi} &= I_y\left(\dot{\psi}s\phi c\theta + \dot{\theta}c\phi\right)\left(\dot{\psi}c\phi c\theta - \dot{\theta}s\phi\right) + I_z\left(\dot{\psi}c\phi c\theta - \dot{\theta}s\phi\right)\left(\dot{\psi}s\phi c\theta + \dot{\theta}c\phi\right) \\
\frac{\partial T}{\partial \dot{\phi}} &= I_x\left(\dot{\phi} - \dot{\psi}sin\theta\right) \\
\frac{\partial V}{\partial \phi} &= \int ydm(-gcos\phi cos\theta) + \int zdm(gsin\phi cos\theta) \\
\frac{\partial V}{\partial \dot{\phi}} &= 0
\end{aligned}
\tag{48}
$$

Considering equation (32) the Lagrangian are:

$$\frac{\partial L}{\partial \phi} = I_y\left(\dot{\psi}s\phi c\theta + \dot{\theta}c\phi\right)\left(\dot{\psi}c\phi c\theta - \dot{\theta}s\phi\right) + I_z\left(\dot{\psi}c\phi c\theta - \dot{\theta}s\phi\right)\left(\dot{\psi}s\phi c\theta + \dot{\theta}c\phi\right) + \int ydm(gcos\phi cos\theta) - \int zdm(gsin\phi cos\theta)$$

$$\frac{\partial L}{\partial \dot{\phi}} = I_x\left(\dot{\phi} - \dot{\psi}sin\theta\right) \tag{49}$$

Introducing equation (49) in equation (35), the result is:

$$\Gamma_\phi = \ddot{\phi}I_x - \ddot{\psi}s\theta I_x - \dot{\theta}\dot{\psi}c\theta\left[cos(2\phi)\left(I_y - I_z\right) + I_x\right] - \dot{\psi}^2 cos^2\theta\frac{sin(2\phi)}{2}\left(I_y - I_z\right) - \dot{\theta}^2\frac{sin(2\phi)}{2}\left(I_z - I_y\right) - \int ydm(gcos\phi cos\theta) + \int zdm(gsin\phi cos\theta) \tag{50}$$

If the angular body rates in equation (10) are derived in time, and introduced in the above formula, the equation can be greatly simplified:

$$\Gamma_\phi = I_x\dot{p} - \left(I_y - I_z\right)\dot{q}\dot{r} - \int ydm(gcos\phi cos\theta) + \int zdm(gsin\phi cos\theta) \tag{51}$$

As it was done with the classical method, the hypothesis of small angles is applied. Following the same process for the angles theta and psi, the equation of motion can be written as:

$$\begin{aligned}\Gamma_\phi &= I_x\ddot{\phi} - \left(I_y - I_z\right)\ddot{\theta}\ddot{\psi}\\ \Gamma_\theta &= I_y\ddot{\theta} - \left(I_z - I_x\right)\ddot{\phi}\ddot{\psi}\\ \Gamma_\psi &= I_z\ddot{\psi} - \left(I_x - I_y\right)\ddot{\phi}\ddot{\theta}\end{aligned} \tag{52}$$

The non conservative moments that were developed in section 2.2.3 are the thrust imbalance of the rotors in the different axis (15) and the gyroscopic torques (17). These terms are added and introduced in equation (52), getting the same dynamic equations as the ones obtained with the classical method.

$$\begin{aligned}\ddot{\phi} &= \frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi} + \frac{J_r\dot{\theta}}{I_x}\cdot\left(\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4\right) + \frac{lb}{I_x}\left(\Omega_4^2 - \Omega_2^2\right)\\ \ddot{\theta} &= \frac{I_y - I_z}{I_y}\dot{\theta}\dot{\psi} + \frac{J_r\dot{\phi}}{I_y}\cdot\left(-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4\right) + \frac{lb}{I_y}\left(\Omega_3^2 - \Omega_1^2\right)\\ \ddot{\psi} &= \frac{I_x - I_y}{I_z}\dot{\phi}\dot{\theta} + \frac{d}{I_z}\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right)\end{aligned} \tag{53}$$

### 2.2.6 Quadrotor kinematics

In this section, the equations that relate the position with the attitude and thrust are obtained. The velocity conversion between body axis and inertial axis can be express as:

$$\begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{pmatrix} = R_{Ab} \cdot \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{54}$$

This equation is derived and the derivation of the rotation matrix is neglected:

$$\begin{pmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{pmatrix} = R_{Ab} \cdot \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} + \underbrace{\dot{R}_{Ab}}_{0} \cdot \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{55}$$

The equation (23) is introduced but neglecting Coriolis terms.

$$\begin{pmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{pmatrix} = R_{Ab} \cdot \left( \underbrace{\begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix}}_{0} + \frac{1}{m} \cdot \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \right) \tag{56}$$

The motor's force in body axis is given by $U_1\vec{k}$ and the gravity force has already been expressed in the inertial frame in (18). If these terms are replaced in the above equation, equations of the quadcopter's position are obtained.

$$\begin{aligned} \ddot{X} &= \frac{U_1}{m} \cdot (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) \\ \ddot{Y} &= \frac{U_1}{m} \cdot (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) \\ \ddot{Z} &= \frac{U_1}{m} \cdot (\cos\phi \cos\theta) - g \end{aligned} \tag{57}$$

## 2.3 Quadcopter model in Simulink

A model is a set of mathematical equations that are used to describe the relationship between input and output of the system. It is unlikely that the model can accurately represent the relationship between the input and output of a system. At any stage of modeling, a number of effects are neglected to make the problem tractable. However, it is still useful to use models that can approximate reality, allowing a number of studies, such as simulation (obtaining numerical or graphical form of the response of a system to a known excitation), control (regulation output of a system to be equal or similar to a given input) or prediction (obtaining the response of a system in the future, various input conditions assumed).

The physical model of the quadrotor obtained in the previous section has been written down in Simulink. The objective is to have a model resembling the real dynamics of the quadcopter.

Using this model, the different controllers that will be built in section Chapter 3 can be regulated and tested in a simple way before being introduced into the embedded system. The model has the following form:

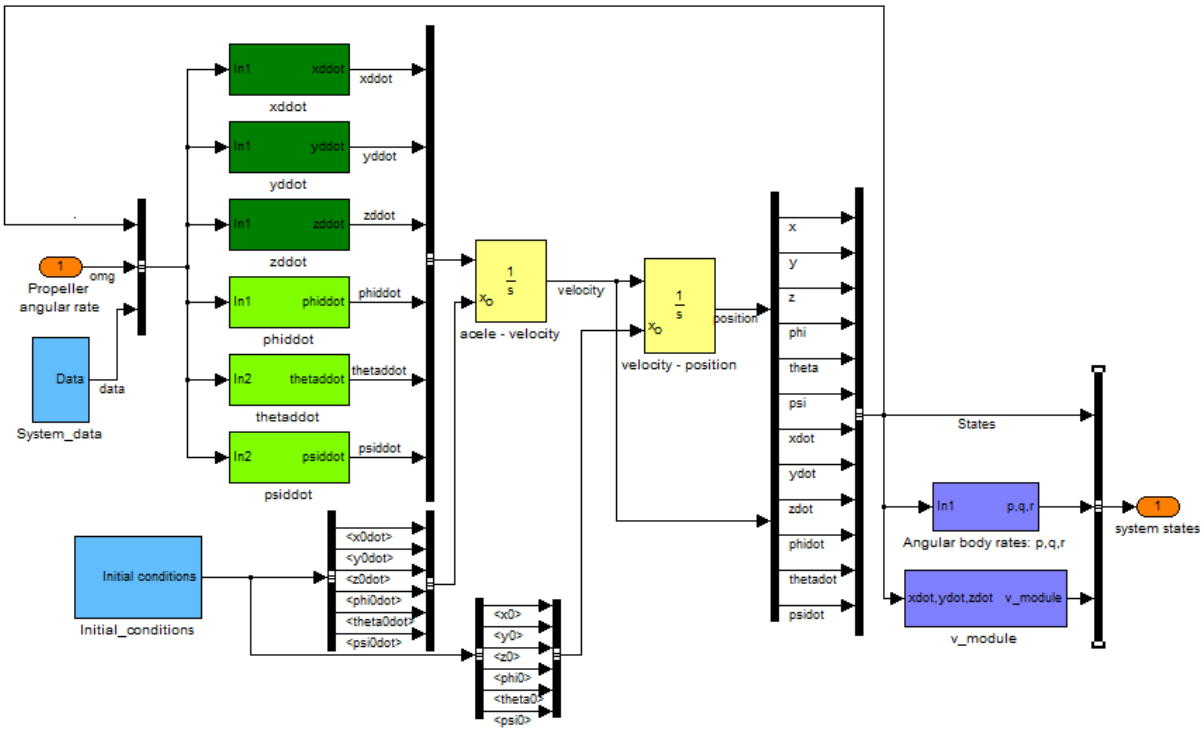



Figure 2.3-1: Quadcopter model in Simulink

The four inputs of the system are the angular rates of the propellers. Also quadcopter constants (mass, moments of inertia about the axis, thrust and drag factor, rotor inertia and the distance between the z axis and the rotors) are introduced into the system. The values of these constants will be calculated in section 2.4. Green subsystems contain the equations of position and attitude (53) and (57) previously introduced. These equations provide the accelerations of the position and angles. In order to obtain the position, the result will be integrated twice. For the integration, the initial values of the positions and angles and its derivates need to be introduced. Also, the equations in green need some values of the system outputs, the angles and their derivation in time. They are introduced into the equations by means of a feedback.

Angular body rates have also been calculated from the system state with the introduction of the equations (10). The velocity module has been calculated with:

$$|v| = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} \tag{58}$$

Next, it will be checked that with the equations calculated in section 2.2 and introduced in Simulink, the system behaves as predicted in section 2.1.2. For this purpose, the speed of each rotor is introduced and the system state is represented.

- Vertical movement

As it has been already seen, vertical motion of the quadcopter is achieved by increasing (or decreasing) the rotational speed of the four rotors in the same proportion. To know from which value, the quadcopter changes its motion from upwards to downwards or vice versa, the hovering condition needs to be found. For this purpose, $\ddot{z}$ in equation (57) is equaled to zero.

$$0 = \frac{4b\Omega^2}{m} \cdot \underbrace{(\cos\phi \, cos\theta)}_{0 \; in \; hover} - g \tag{59}$$

$$\Omega = \sqrt{\frac{mg}{4b}} = 152.64 \, rd/s \tag{60}$$

This value is introduced of radians per second is introduced in each one of the four entries. Introducing an initial altitude of 2 meters, it is obtained:

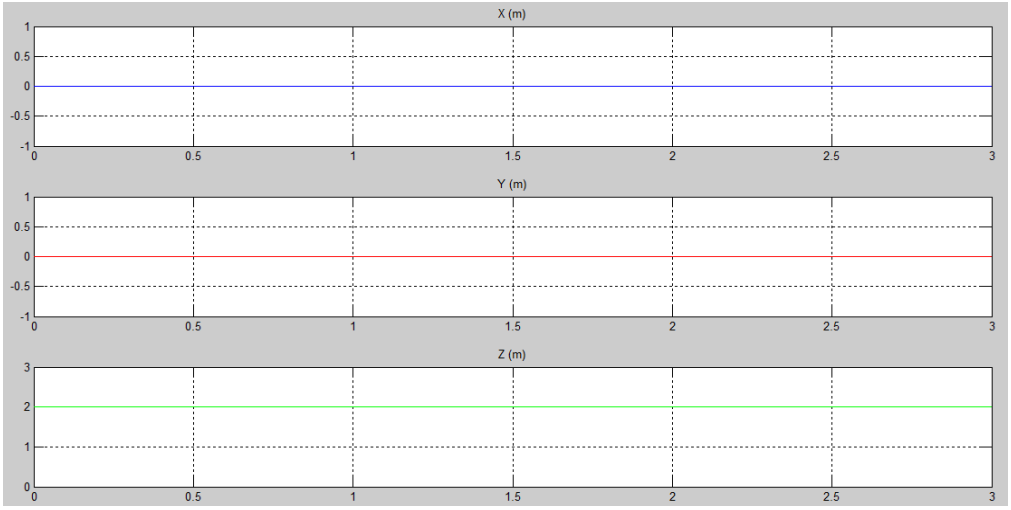



Figure 2.3-2: Position for hovering

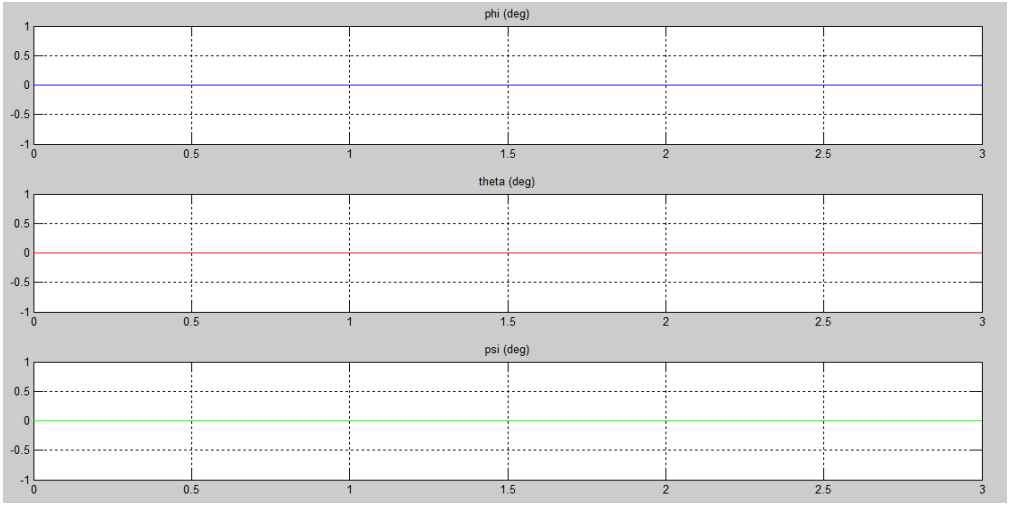



Figure 2.3-3: Attitude for hovering

All the values are cero and the quadcopter remains in its initial position. The angular body rates and velocities are also zero. If a higher value of angular speeds is introduced, the

quadcopter ascends maintaining its attitude to zero. In the following graphs, 154rd/s have been introduced in the propellers, with an initial altitude equal to zero.

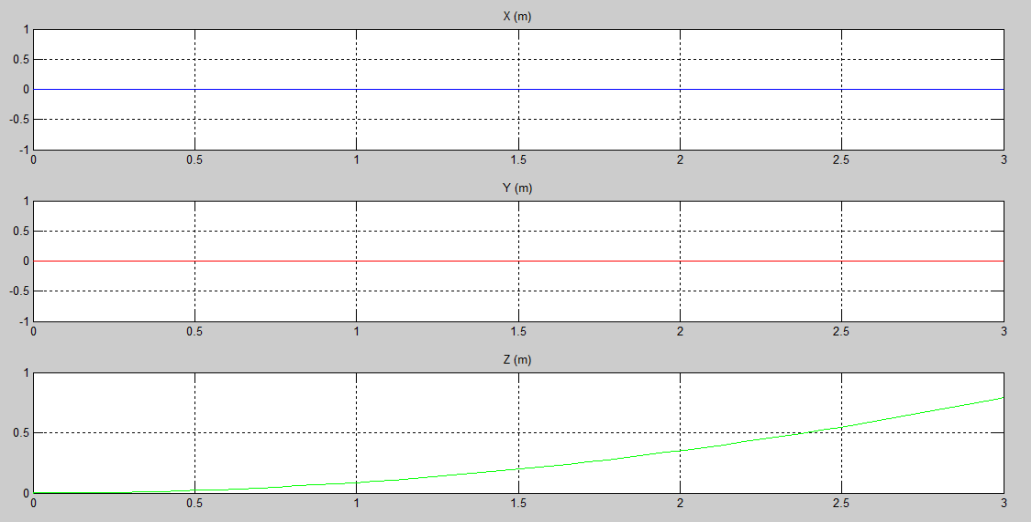



Figure 2.3-4: Position for vertical movement

As it can be seen above, the quadcopter ascend with a constant acceleration of 0.175 m/s$^2$. The attitude angles have not changed from its initial position. If they value introduced is smaller than the hovering value, the acceleration value is negative and the quadrotor descends.

- Right and left movement

To move the aircraft to the left or what is the same, to the positive values of y, is needed to increase thrust of motor 2 and decrease 4. The values used for the simulation are: $\Omega_2 = 152.74$, $\Omega_4 = 152.54$rd/s and the same values as before for the other two motors.

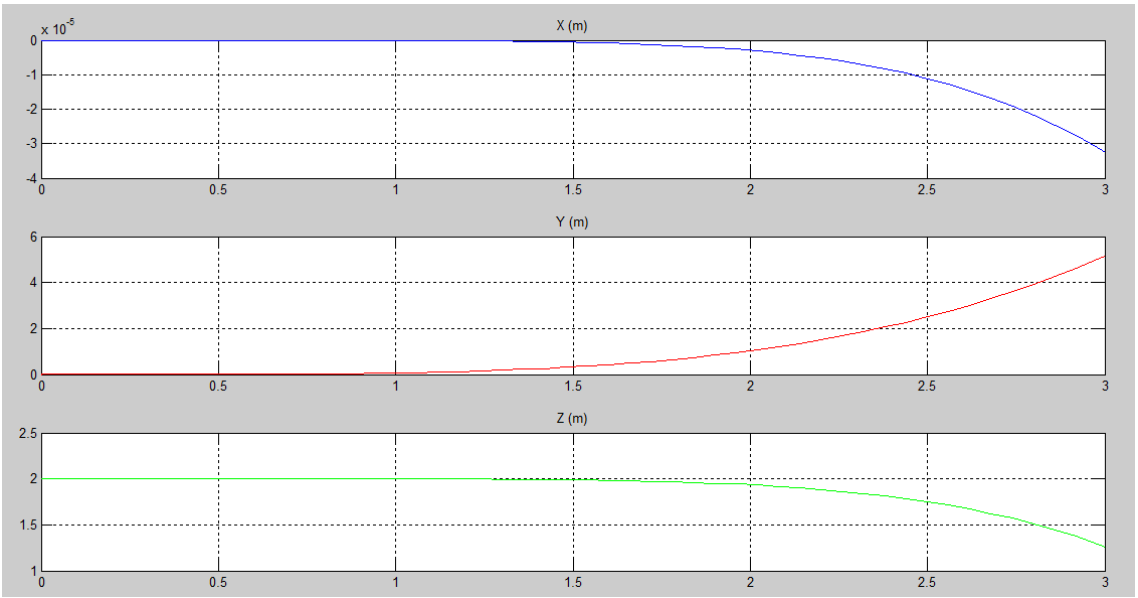



Figure 2.3-5: Position for motion to the left

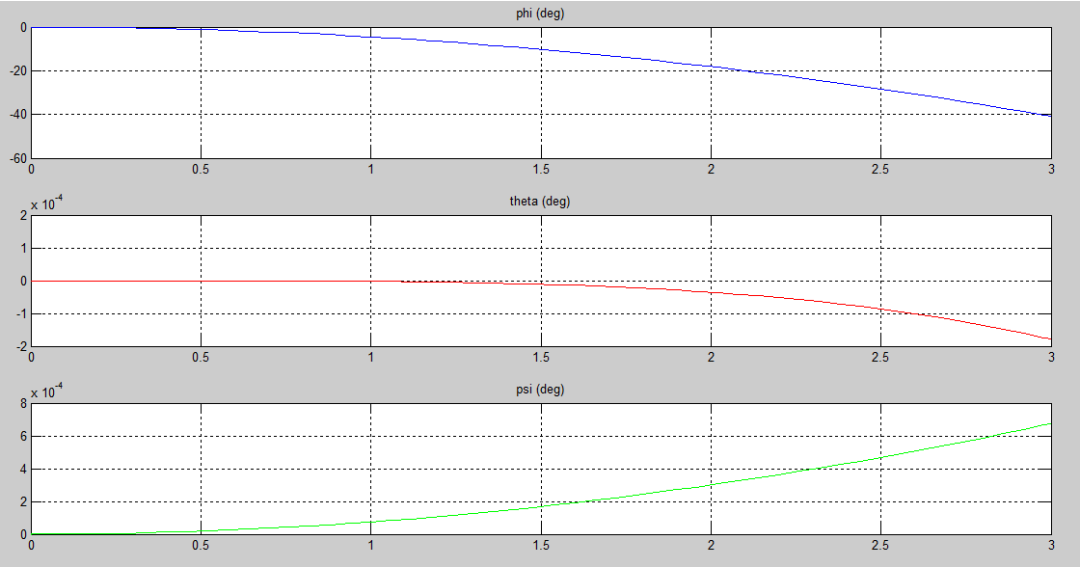



Figure 2.3-6: Attitude for motion to the left

The quadrotor moves as expected. Its altitude decreases because as now it is tilted, the vertical thrust that helps to offset the weight is lower. To augment this thrust, the propellers angular rates should be higher. The value of the angles θ and $\psi$ are not completely zero due to the coupling between the equations but are much smaller than the other angle.

- Forward and backward movement

To proved that the behavior in the direction x is the appropriate, the same procedure as the above section has been done. The values used for the simulation are: $\Omega_3 = 152.74$, $\Omega_1 = 152.54$ and $\Omega_2 = \Omega_4 = 152.64$ rd/s. With this propeller angular rates, the quadrotor moves in x axis and the angle theta is positive as expected.

- Yaw movement

To make the vehicle rotate about the z axis, the opposite motors have the same speed. The values introduced are going to be higher for the motors that turn clockwise: $\Omega_2 = \Omega_4 = 152.74$ rd/s and $\Omega_1 = \Omega_3 = 152.54$ rd/s. The position values are zero and the angles are the followings:

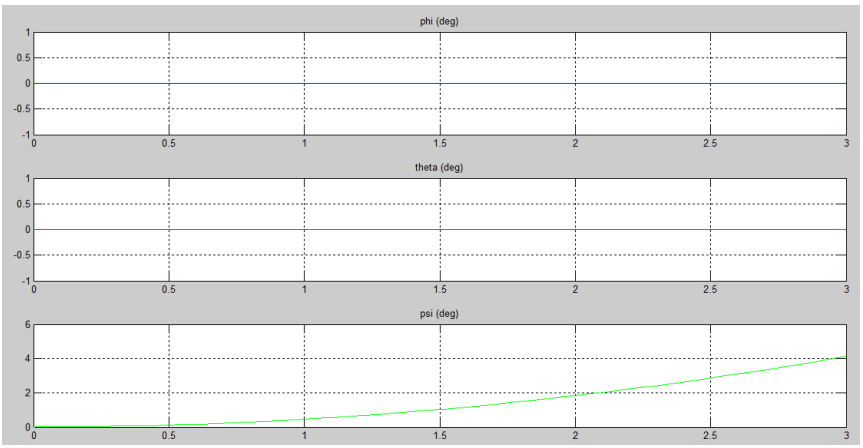



Figure 2.3-7: Attitude for yaw motion

## 2.4 Identification of the constants

Different constants need to be implemented in the model in order to be able to simulate it successfully. Data information of the quadricopter's different systems is not as accurate and bounteous as it should. For this reason, some of the techniques that were used to obtain them are not optimal. The constants can be divided into the following categories: basic measurements, moments of inertia and propeller and motor constants.

### *2.4.1 Basic measurement*

This section will be focus in the measurement of the masses and lengths. The quadcopter was divided in different components that were weighted with a balance. Lengths were obtained from the manufacturer's specifications. Other lengths were taken approximately in order to create a model in Catia. This model will permit to calculate the moments of inertia as well as the mass center in an easy way. This CAD model is not very detailed, the components have a simplified geometry and the smaller ones have been grouped to be represented.

In Figure 2.4-1, picture of the frame and rotor of the real quadcopter can be observed. Figure 2.4-2 shows the different modules of the model.

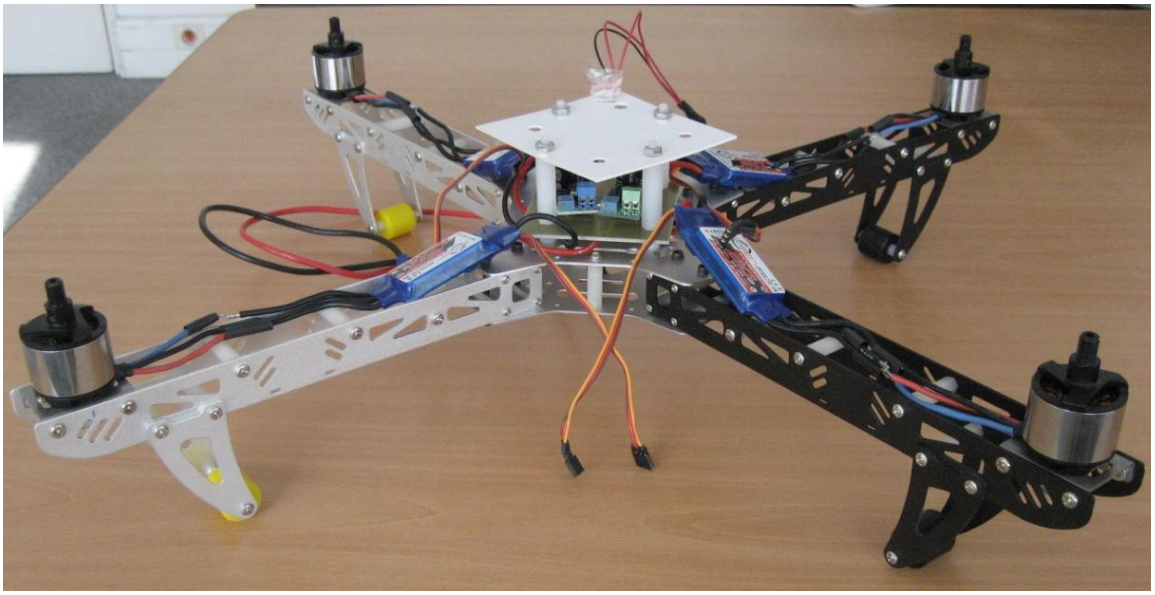


**Figure 2.4-1 Real quadcopter**

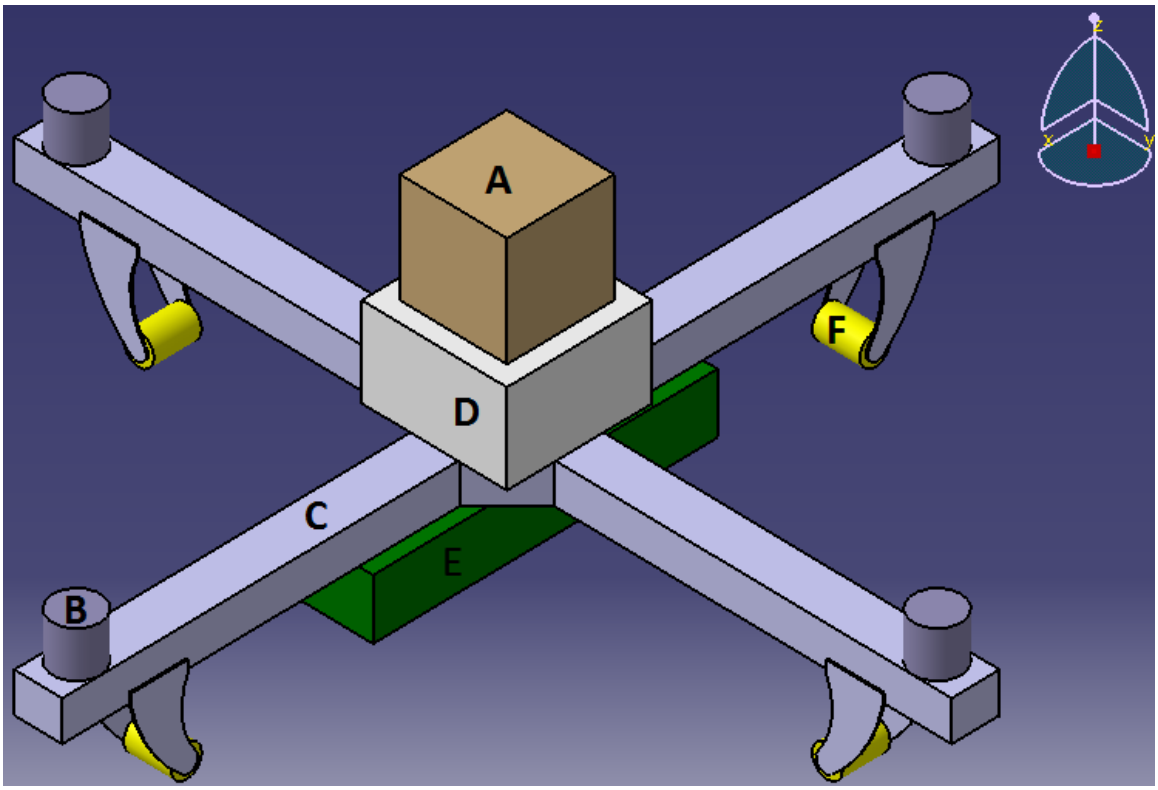



Figure 2.4-2 Quadcopter's modules

- Part A: GPS, WiFi card and antennas

This part includes a WiFi shield with an update rate up to 2 Hz, a GPS shield and the antenna. It has a mass of 110 grams and has been modeled as a rectangular prism center in the z axis of the quadcopter.

- Part B: Propellers and motors

Brushless motors with electronic control for model airplanes are used. Unlike a conventional DC motor, brushless motors have no armor motions and it is the permanent magnet the one that rotates on its axis. Each propeller has a mass of 12.5 grams and each motor weighted 50 grams. They both are modeled as a cylinder of the dimensions of the rotor which are given by the manufacturer: 28mm in diameter and 26 in height. The distance between the z axis and the propeller´s axis is also given by the manufacture. This length is going to be called l, since it appears in a lot of formulas of the model.

$$l = 22{,}5\ cm$$

- Part C: Frame

This part has been model as a solid cross with a square in the middle. The union between the frame and the supports has been drawn just to make the design clearer since its mass is very small to be taken into account. The mass of the frame is 340 grams.

- Part D: FlyMaple card and power regulator

The FlyMaple card is a quadcopter controller board that incorporates a microcontroller and an inertial measurement system that includes:

- 3-axis gyroscope: it detects changes in pitch, roll and yaw using the Coriolis effect.
- 3-axis accelerator: It detects the current rate of acceleration
- 3-axis magnetometer: They are not always included and are used mostly to allow better performance for dynamic orientation calculation
- a barometric pressure sensor: used to calculate the altitude

This block has a mass of 100 grams and has been modeled as a rectangular prism center in the z axis of the quadcopter.

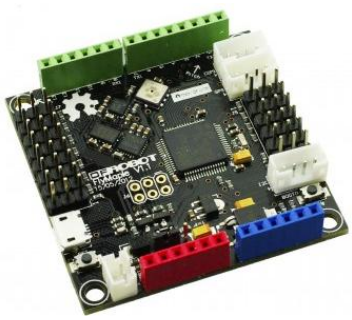


Figure 2.4-3 FlyMaple card

- Part E: Battery

The battery is Lithium-Polymer type and is used for applications where high power and low weight are required. The battery has a rectangular form, as it has been modeled in Catia. It has a large mass compared to the rest of the elements and it is situated mainly in the x axis. This is going to cause different inertia moments in the axis x and y as we will see below. The battery's mass is 300 grams.

- Part F: Supports

These are the parts of the quadcopter that are in touch with the ground and they are made of plastic material. The support of the quadcopter has been design separately although due to its small mass, it's not going to affect the global result significantly. The mass of the four of them is 40 grams.

Adding the masses of the different parts, we obtain a mass of:

$$m = 1.14\ kg$$

The battery and the four motors are the more significant parts of the quadcopter's mass. An even simpler model could have consisted in four cylinders situated in the position of the rotors and given them a mass equal the sum of the masses of the rotors and a part of the mass of the frame. A cube with the mass of the battery and the rest of the elements would be in the middle.

### 2.4.2 Moments of inertia

Once all the masses, positions and volumes of all the systems are known, the program Catia will calculate the mass center and the inertia axes of each part. The results given by the program were:

| | Mass | $Z_{mc}$ | $I_x$ | $I_y$ | $I_z$ |
|---|---|---|---|---|---|
| A | 110 | 72,5 | 6,50E-04 | 6,50E-04 | 0,001 |
| B | 250 | 13 | 0,006 | 0,006 | 0,013 |
| C | 340 | -10,36 | 0,003 | 0,003 | 0,007 |
| D | 100 | 22,5 | 6,49E-05 | 6,49E-05 | 9,61E-05 |
| E | 40 | -60 | 6,50E-04 | 6,50E-04 | 0,001 |
| F | 300 | -35 | 6,25E-05 | 8,32E-04 | 8,50E-04 |

**Table 1: Masses and inertia axis of the quadcopter**

To calculate the global moments of inertia it's enough to add the moments of each part of the quadcopter:

$$Ix = 1.04 \cdot 10^{-2} kg \cdot m^2$$
$$Iy = 1.12 \cdot 10^{-2} kg \cdot m^2 \quad (61)$$
$$Iz = 2.29 \cdot 10^{-2} kg \cdot m^2$$

In order to find the global mass centre in the z axis, we employ the following equation:

$$Z_{CM} = \frac{\sum m_i \cdot z_i}{M_t} \quad (62)$$

The term $M_t$ is the total mass, $m_i$ is the mass of each part of the system and $z_i$ the centre mass's position of each part in the z axis. With all the assumptions that have been taken in the design of the model, the solution is that the centre of mass is 2.58 mm below the body fixed frame. However, as it has been seen in section 2.1.3 they are supposed to be coincident.

### 2.4.3 Propeller and motor constants

There are three parameters of the motor and propellers that need to be calculated because they have a very important role in the dynamics of the quadcopter, as it's already seen. These parameters were the thrust coefficient (b), the drag coefficient (d) and the rotor's inertia.

- Thrust coefficient

The thrust generated by the engine and propeller assembly is used to control the roll and pitch motion. This force has the form: $F_i = b\Omega^2$, where $F_i$ is the force in Newton of each propeller and

Ω is the propeller speed in radians per second. The most appropriate way to calculate this value would be with a test as in (Agarwal, 2012). It would calculate the thrust generated by the propellers in static conditions. The motor would be bolted to a heavy block which would be place on a weighing machine. The range of RPM would be widely varied and the values in the reading of the weighing machine would change. These values are directly related with the propeller's thrust.

Since the material to do this experiment is not available, the program "Drive calculation" has been used to calculate this value. The dimensions of the propeller, which are 10 inches in length and 3.8 in pitch, are introduced. As to the characteristics of the rotor, it is of the 2212 series, it weights 50 grams, a KV (rev/V) of 2200, its maximum power is 342W and it supports a battery of 2-3Li-Po. The closest engine, containing the program, has been introduced. A direct drive gearbox has also been selected since the rotor's rotation is the same as the propeller. The results of the program are shown in the following picture:

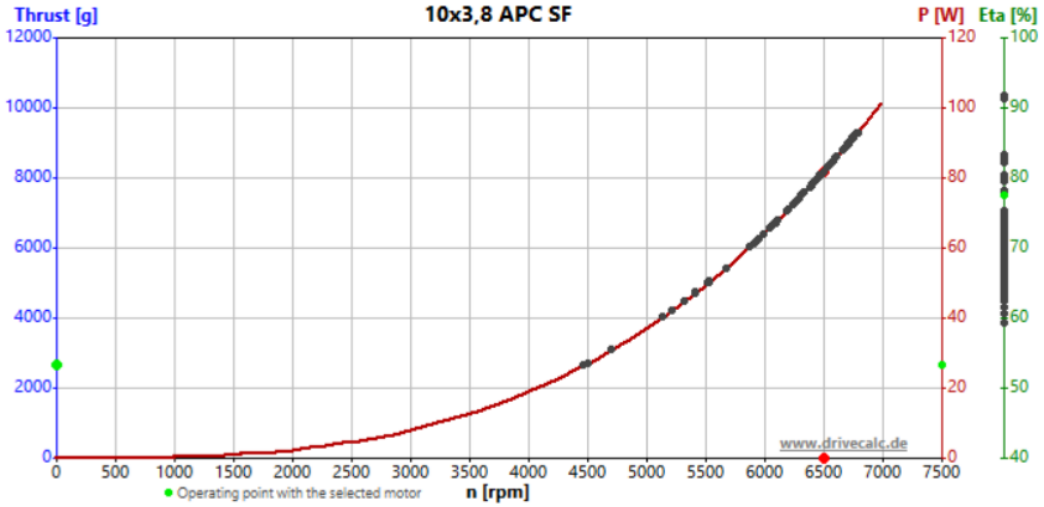



Figure 2.4-4 Estimation of b constant

Thrust units have been converted in Newton and the rpm in radians per second. Using the formula $b= F_i/\Omega^2$ for different values and taking the average, the b value that will be used is:

$$b = 1.2 \cdot 10^{-4}\ Ns^2$$

- Drag coefficient

The drag's coefficient appears in the yaw control, as it is shown in the following equation:

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} + \frac{d}{I_z}\left(-{\Omega_1}^2+{\Omega_2}^2-{\Omega_3}^2+{\Omega_4}^2\right) \quad (63)$$

The experiment that could be performed to calculate this value consists on placing the quadrotor on a platform that allows only the free vertical rotation. A known value of power will be introduced in two of the rotors (1 and 3 or 2 and 4). As yaw is the only rotation

permitted, the first term of the equation is negligible. If the equation is now integrated two times:

$$\psi(t) = \frac{d}{I_z}(2\Omega^2) \cdot \frac{t^2}{2} \quad (64)$$

If the equation (64) is solved for d and the time taken by the drone to give a quarter turn is measured, all the information to calculate the d is available.

$$d = \frac{\pi/_2 \cdot I_z}{\Omega^2 \cdot T^2} \quad (65)$$

However, in this case, this coefficient has been calculated in relation with the b coefficient. In all the literature that has been consulted, such as (K.Alexis, G. Nikolakopoulos and A. Tzes, 2011) or (BOUABDAHALL, 2007), the value of d was approximately a 2.5% of the b value.

$$d = 3 \cdot 10^{-6}\ Nms^2$$

- Rotor's inertia

The rotor's inertia is composed of two constants: the rotational moment of inertia around the propeller axis an around the motor axis. The contribution of the blade can be modeled as a flat plate and the other as a solid cylinder. Their moments of inertia are equal to:

$$J_{propeller} = \frac{1}{12} \cdot M_B \cdot \left({W_B}^2 + {L_B}^2\right) = 2.11 \cdot 10^{-4}\ Kg \cdot m^2 \quad (66)$$

where $M_B$ is the mass of one blade, $W_B$ is its width and $L_B$ its longitude. The solid cylinder has the following equation, where $M_m$ is the mass of the motor and R the radio.

$$J_{motor} = \frac{1}{2} \cdot M_m \cdot R^2 = 1.96 \cdot 10^{-5} Kg \cdot m^2 \quad (67)$$

The rotor's inertia can be calculated as:

$$J_r = J_{motor} + \frac{J_{propeller}}{4} = 7.25 \cdot 10^{-5} Kg \cdot m^2 \quad (68)$$

### *2.4.4 Model constrains*

The increase of the rotational speed generates an augmentation of the thrust force, as it has been seen. However, this speed cannot be increased to infinity; rotors have a maximum speed value that shouldn't be exceeded in order to maintain their integrity and not shorten its life. This value is going to be introduced in the quadcopter model, as a Simulink saturation module, so the control is done without overtake this situation. As a precise value is not available, it has been supposed that the maximum speed value is the one that permits to generate a thrust that has twice the value of the quadcopter's weight as it is done in (Abolghasem Naghash,

Mehrdad Naghshineh and Ali Honari, 2013). This constrained value is computed in the following equations.

$$T = 4b\Omega_{max}{}^2 \tag{69}$$

Substituting the values that were found in the previous section and knowing that T= 2mg it is obtained:

$$\Omega_{max} = \sqrt{\frac{2mg}{4b}} = \sqrt{\frac{2 \cdot 1.14 \cdot 9.8}{4 \cdot 1.2 \cdot 10^{-4}}} = 215.75\ rd/s \tag{70}$$

The saturation speed of the propeller is a 41.4% higher than the hovering speed. The rotor speed during the simulation will have values between 0 and 215.75 rd/s.

If this value is introduced in the equations (15), the maximum thrust and torques of the quadrotor are:

$$\begin{aligned} U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) = 4b\Omega_{max}{}^2 \rightarrow 0 \leq U_1 < 22.34\ N \\ U_2 &= l \cdot b(\Omega_4^2 - \Omega_2^2) = lb\Omega_{max}{}^2 \rightarrow -1.257 \leq U_2 < 1.257\ Nm \\ U_3 &= l \cdot b(\Omega_3^2 - \Omega_1^2) = lb\Omega_{max}{}^2 \rightarrow -1.257 \leq U_3 < 1.257\ Nm \\ U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) = 2d\Omega_{max}{}^2 \rightarrow -0.279 \leq U_4 < 0.279\ Nm \end{aligned} \tag{71}$$

# Chapter 3

# System Control

The objective of control is to produce a desired result. It includes all the artificial and non artificial processes. Within these last there are the brain that controls heart rate, body temperature and blood pressure; and temperature variation of the sun that controls the metabolism of living beings. Control design for an underactuated artificial system, as the quadcopter presents a great challenge to the automatic area. This problem is considerably increased due to uncertainties that are usually present and sometimes significant, like the un-modeled dynamics, external disturbances, parameter estimation errors and noise.

The controllers are used to simplify the operators' labor and to exert a better control of the operations. One of the objectives of any control system is the stability. A system is stable when after some time, its response (output) remains constant or reaches a limit cycle. In the case of a constant output, this time is called settling time and the achieved value is the steady-state value. The settling time is usually defined as the time from which the error is less than 2%. The maximum overshoot is the maximum peak value of the response curve measured from the desired response of the system. A system is considered unstable when its response after a set time remains oscillating, varying between a periodical range of values or just any random value is obtained. In the following graphic this variables are represented for a step input.

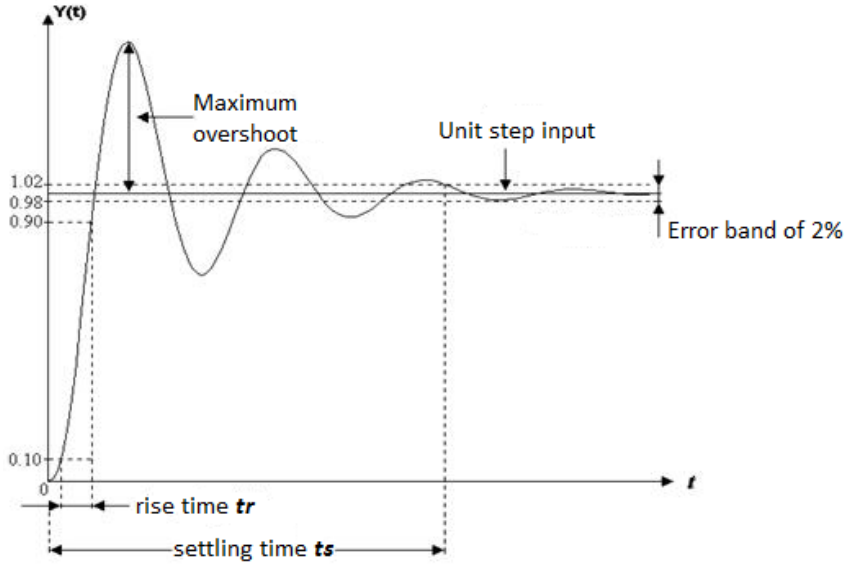



Figure 2.4-1: Response parameters

Another important feature is the speed answer. It shows how quickly the system can reach the value of steady state. The figure shows the types of responses that can be obtained depending on the response speed.

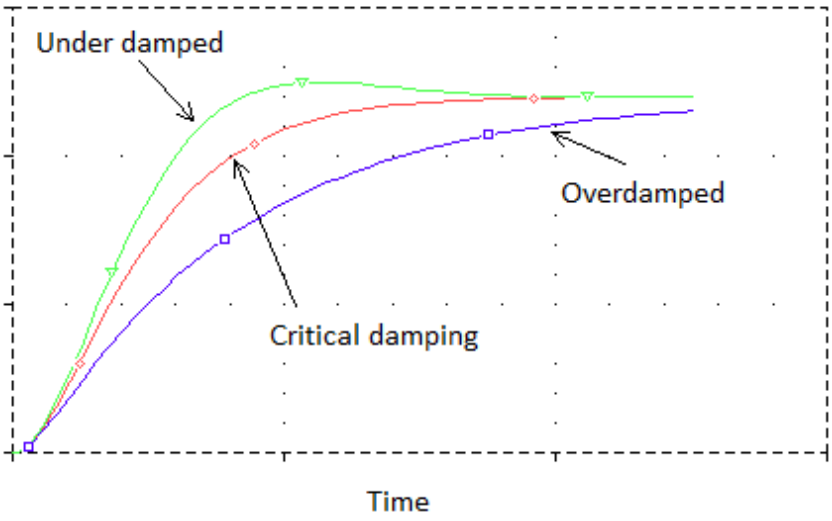



Figure 2.4-2: Types of responses

Each subsystem has a mathematical model. The plant is the system that needs to be controlled and the controller is the system that controls the plant. The open-loop control has success only if the controller has beforehand an accurate knowledge of the plant's behavior, since there is no hint of the output. This control is used in systems that are simple, its accuracy depends on the previous calibration of the system and the stability after a perturbation is not assured.

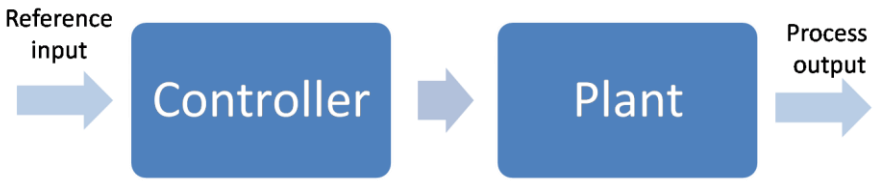



Figure 2.4-3: Open loop control system

In closed-loop control systems the difference between the actual output and the desired output is feedback to the controller to meet the desired system output. This output must be equal to the reference input as soon as possible. Obtaining the transfer function of the controller is the main task. The feedback has the following effects:

- Reduces the sensitivity to variations of the plant
- Reduces the sensitivity to external disturbances
- Allows to control the frequently response
- Allows to control the transient response

- Allows to control the permanent error
- Allows to modify the system stability
- Makes the control more expensive

Feedback control systems are used to control temperatures, level in tanks, pressures or fluxes. Some examples of their used are the elevators, showers, refrigerators and the cruised control of a car.

The general structure of a closed-loop feedback control system is described in Figure 2.4-4.

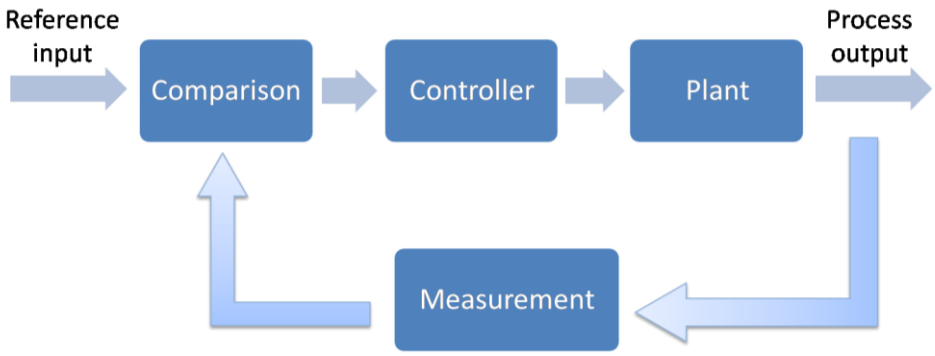



Figure 2.4-4: Closed loop control system

The output of the process is a physical value. For the effectiveness of the control, it is needed that the measurement instruments represent this physical value in a useful way (precise, accurate, reliable, reproducible and in time).

The quadcopter is hard to control not only because its nonlinearity but also because it is underactuated. This means that six degrees of freedom (x, y, z, $\phi$, θ, $\psi$) are controlled with only four control signals: three for torques and one for the total thrust. As a result, it is not possible to control all of the states at the same time. However, a good controller should be able to reach the desired position (x, y, z) with a desired yaw angle ($\psi$) while keeping the stabilization of the pitch (θ) and roll ($\phi$) angles.

Since the quadrotor is very unstable, it is not likely to be flown in open loop. Corrections have to be made very often. For this reason, only close-loop flying is feasible. In this section, three different controllers are going to be implemented in the system. The first one is a PID that will control the attitude and altitude of the quadcopter. The second developed controller is a Integral Backstepping (IB) controller that permits to control the 3D position of the vehicle. The last controller is more sophisticate and it is comprised of a nonlinear $H_\infty$ to control the attitude in the inner loop and an integral predictive controller in the outer loop.

## 3.1 Control using PID

The PID was one of the earliest control methods to be implemented and it is the most used linear regulator in the industry. Even thought there are different algorithms that provide a better performance than PID, the last ones are chosen due to its simple structure, good performance in several processes and tunability even without a specific model of the system.

A PID (Proportional Integral Derivative) is a feedback control mechanism that calculates the deviation or error between a measured value and the value to be obtained, to implement corrective action to adjust the process. The calculation algorithm of PID control is given in three different parameters: the proportional, integral, and derivative. The response of the controller can be described in terms of controller response to an error, the degree to which the controller reaches the set point, and the degree of system oscillation.

The differential equation for a PID controller has the following form:

$$U_{PID} = K_p \cdot e + K_i \cdot \int e\, dt + K_d \cdot \frac{de}{dt} \tag{72}$$

where the value *e* is given by:

$$e = desired\ signal - actual\ signal$$

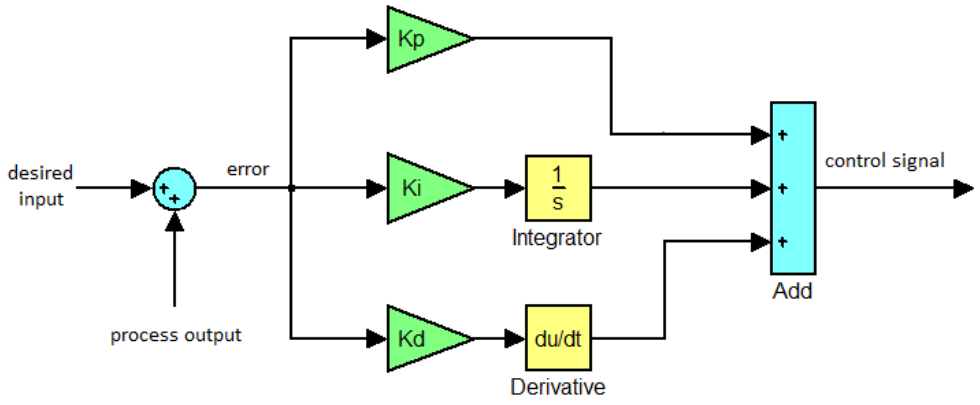



Figure 3.1-1: PID scheme

The values $K_p$, $K_d$ and $K_i$ are three constants that must be chosen depending on the desired behavior. This algorithm has one input and one output. The entry is the error obtained from the difference between the desired reference signal and the signal that is being received. The output is the control signal; in this case, the power that must be applied to the motor to correct the error. The mentioned output is generated through the 3 parts composing the algorithm, the proportional part, integrative and derivative (Basta, 2012).

- The proportionate part is calculated by multiplying the input error by the constant $K_p$. The proportional value determines the reaction of the current error. This type of controller can be seen as an amplifier with an adjustable gain. There is a limit to this

value, beyond it, the system reaches values higher than desired, which is known as overshoot.

- The integral part generates a correction which is proportional to the integral of the error. It is calculated by adding the errors that have been getting in order to compensate the error until it is cancelled. It increases the maximum overshoot and the settling time, the rise time is decreased. Applying a sufficient control effort, the tracking error is reduced to zero. However, the systems are usually more oscillatory, if not adjusted properly can become unstable.

- The derivative term reduces the overshoot and the steady state time. The steady state error is not affected unless it varies with time. A common problem that has the PID is that in the derivative part, noise can affect the system. These small changes can greatly affect the output.

| | Rise time | Overshoot | Settling time | Steady State error |
|---|---|---|---|---|
| Kp | Decrease | Increase | Small change | Decrease |
| Ki | Decrease | Increase | Increase | Eliminate |
| Kd | Small change | Decrease | Decrease | Small change |

Table 2: Behavior of the PID constants

Some applications may not require the three controllers, a PID can also be called PI ($K_d$=0), PD ($K_i$=0), P or I in the absence of the respective control actions.

For readability and physical sense, it is preferable to write the formula as follows:

$$U_{PID} = K_p\left(e + \frac{1}{T_i}\cdot\int e\,dt + T_d\cdot\frac{de}{dt}\right) \quad (73)$$

where Td is the derivation time constant and Ti the integration time constant. In this manner, with the integral action, the error is integrated over a time interval Ti; and with the derivative one, error is derived after a time Td. A very important part when developing a PID consists of settling these parameters for each command.

### *3.1.1 Implementation in Simulink*

The model was built such that it has 4 inputs which are the desired altitude and the desired attitude of the quadrotor. The Euler angles and the height are used as a feedback to the proposed controller in order to achieve the desired orientation and angles.

An important simplification of the developed model was carried out. The control of the different angles is supposed to be decoupled. With this assumption, three separated plants with a different PID controller are obtained. Moreover, it is clear that the engines cannot

exceed their maximum. Therefore, some saturations blocks have been implemented in the rotor speed and in the total thrust as it was seen in section (2.4.4).

In the following graphs, the different subsystems of the model are represented. Figure 3.1-2 is a representation of the whole system. The four inputs are selected in a signal builder block. The signal is defined in degrees. As the quadcopter model is defined in radians, there is a module that converts the units from degrees to radians. Besides the quadcopter's subsystem that has already been described, there are to other subsystems: the PID controller and the motor speed control.

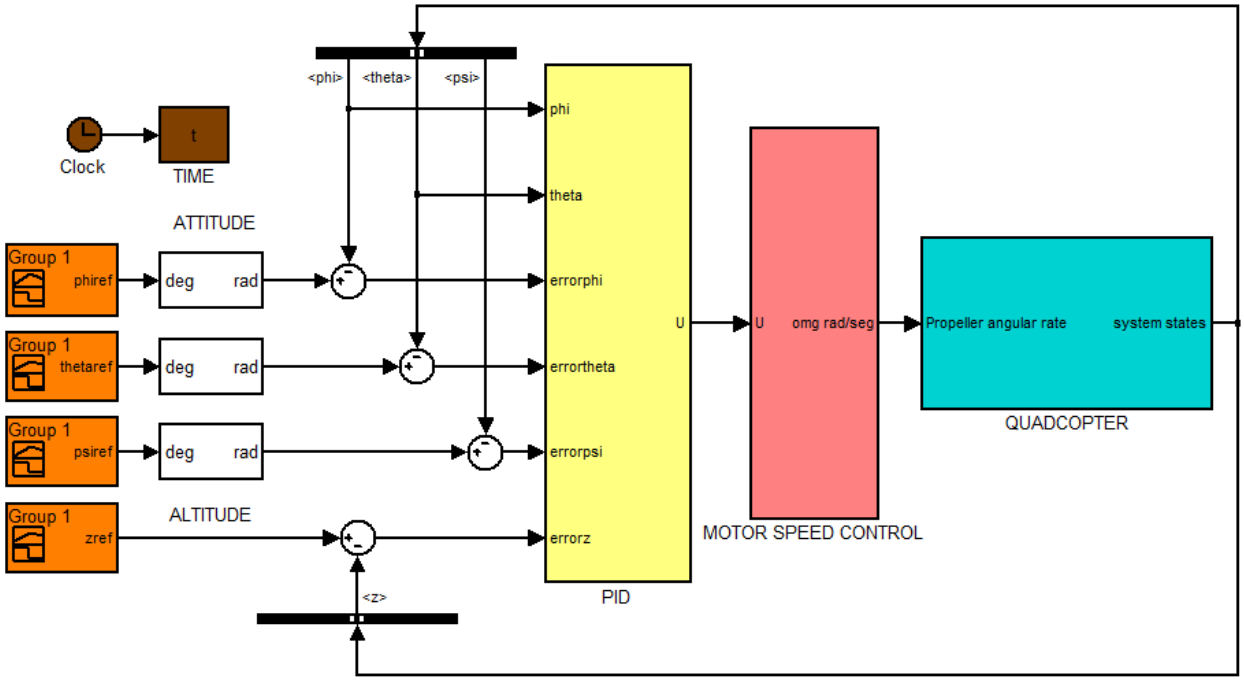



Figure 3.1-2: System in Simulink with a PID controller

The motor speed control subsystem, converts the values $U_1$, $U_2$, $U_3$ and $U_4$ (vertical thrust, rolling, pitching and yawing moments) in revolutions per minute of each rotor. The formulas that are used are the following. They are obtained solving for propeller speeds, the equation (15) in terms of the total thrust and torques.

$$\Omega_1^2 = \frac{1}{4b}U_1 - \frac{1}{2bl}U_3 - \frac{1}{4d}U_4$$

$$\Omega_2^2 = \frac{1}{4b}U_1 - \frac{1}{2bl}U_2 + \frac{1}{4d}U_4$$

$$\Omega_3^2 = \frac{1}{4b}U_1 + \frac{1}{2bl}U_3 - \frac{1}{4d}U_4 \quad (74)$$

$$\Omega_4^2 = \frac{1}{4b}U_1 + \frac{1}{2bl}U_2 + \frac{1}{4d}U_4$$

The equations (74) are introduced in the yellow blocks of the Figure 3.1-3.

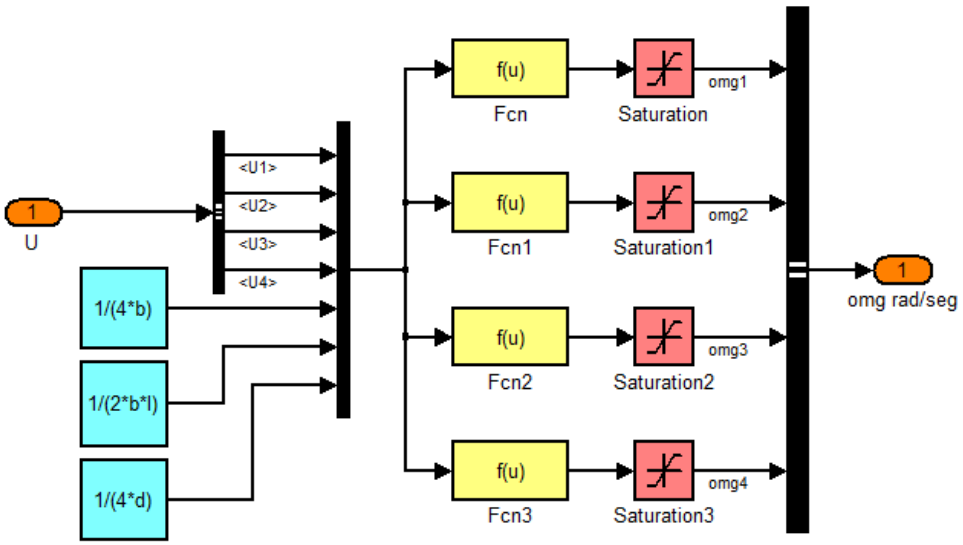



Figure 3.1-3: Motor speed control subsystem

The PID subsystem has as entries the errors of each one of the values that will be control: altitude and the Euler angles (phi, theta and psi). When the attitude errors pass though the PID, the values of U2, U3 and U4 are directly obtained as outputs. However, the output of the altitude PID is not U1. To get U1 the equation (57) in z axis needs to be introduced. In that equation, $U_z$ is the output of the PID and g is the gravity. For this reason, the angles phi and theta are also introduced as entries in the subsystem.

$$U_1 = \frac{m \cdot (U_z + g)}{\cos\phi \cdot \cos\theta} \tag{75}$$

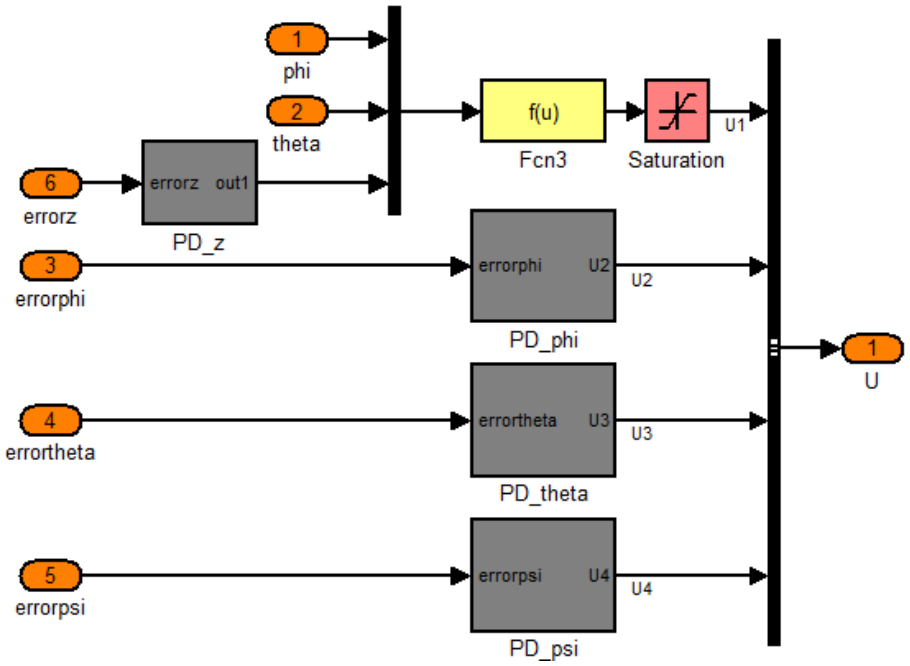



Figure 3.1-4: Subsystem with all the PIDs

The grey parts of the fFigure 3.1-4 are the PID controllers. They have all the same form. Below, one of the controllers is represented.

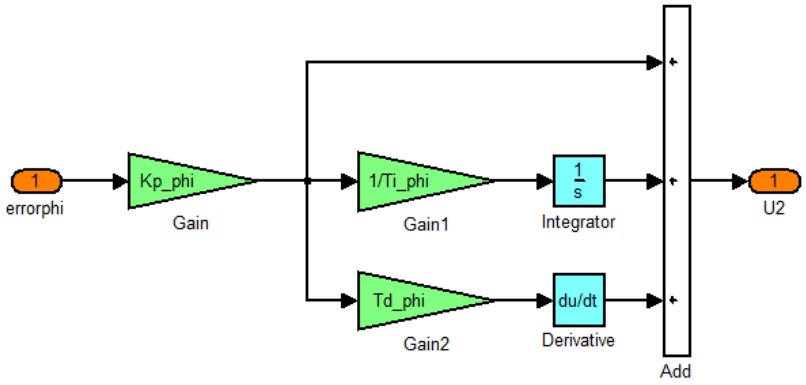



Figure 3.1-5: PID subsystem

The settling of a PID consists of finding the values $K_p$, $T_i$ and $T_d$ to obtain a response which is fast, accurate, without overruns and robust. The last parameter, robustness, means that the regulation always works even if the model changes a little.

### *3.1.2 Simulation results*

In the first PID control applications, the adjustment was based on the operators' own experience. The processes were slow, and each tuning test could take hours or even days. To address these issues, Ziegler and Nichols (1942) proposed empirical techniques to tune PIDs without any knowledge of the plant to be controlled. In this project, their closed-loop method is going to be tested for the tuning.

The objective of this method is obtaining an overshoot lower than the 25% for a step entry. Using only the proportional control ($T_d = 0$ and $T_i = \infty$) increase the gain $K_p$ to go from zero to a critical value at which the system exhibits a sustained oscillation. This value is called the critical gain $K_{pc}$. If the system doesn't have a sustained oscillation for any value of $K_p$, this method is not applicable. The following step is to obtain graphically the period of these oscillations $T_c$. The recommended parameters given by the Ziegler-Nichols formulas are:

| | $K_p$ | $T_i$ | $T_d$ |
|---|---|---|---|
| P | 0,5 $K_{pc}$ | ∞ | 0 |
| PI | 0,45 $K_{pc}$ | $P_c$/1,2 | 0 |
| PID | 0,6 $K_{pc}$ | 0,5 $P_c$ | 0,125 $P_c$ |

Table 3: Constants for the Ziegles and Nichols method

The Ziegler and Nichols method is going to be implemented to the z control. All the entries of the system are zero except the altitude which suffers a step entry of gain one in the first second. Taking into account only this controller, the gain $K_p$ is increase. When the value is equal to 15 ($K_{pc}$=15), $T_d$ zero and $T_i$ infinity; sustained oscillations are obtained.

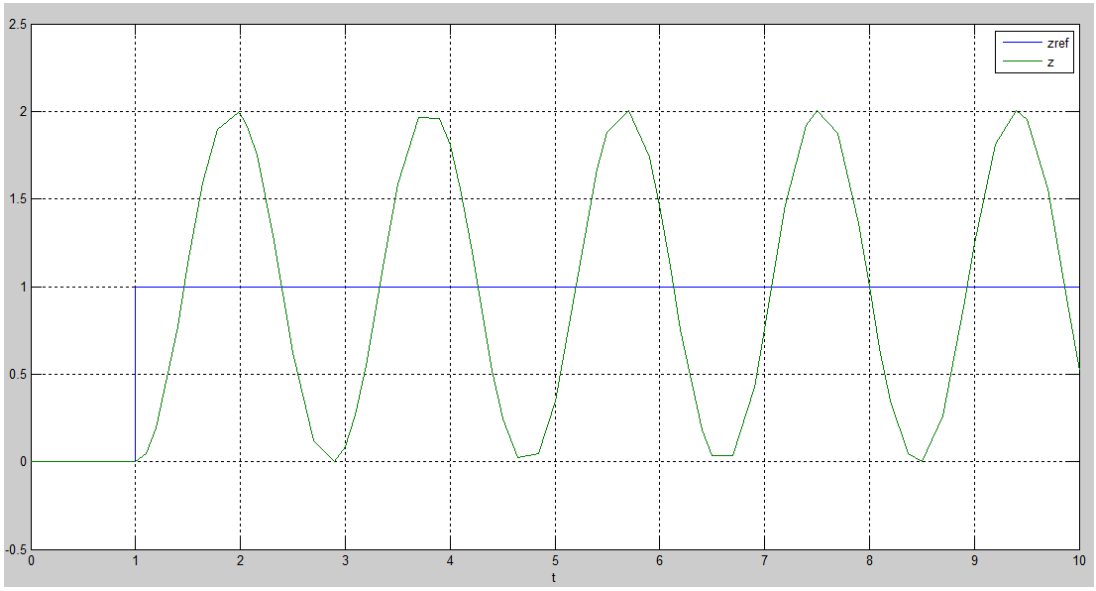



Figure 3.1-6: Ziegles-Nichols method

The period of these oscillations is $T_c$=1.9 seconds. Using the previous graph, the constants using the Ziegler-Nichols method are:

Kp=9
Ti=0.95
Td=0.2375

The response obtained with these constants is represented in the following picture:

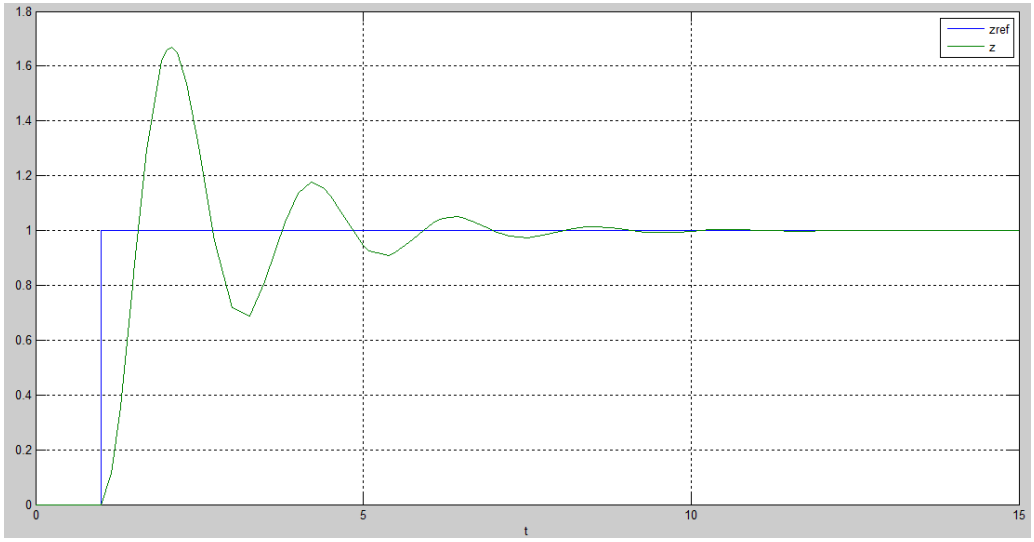



Figure 3.1-7: Step response in z using Ziegles-Nichols method

As it can be seen in Figure 3.1-7 the method is not very satisfactory in this case. The steady-state error is zero but the settling time is very large and the overshoot much higher than the

25%. A manual tuning has been realized in order to obtain better results, taking into account that:

- Adding $K_p$ reduces the time rise but the overshoot is bigger. The settling time changes very little and the settling error is improved.
- If the inverse of $T_i$ is increased, the rise time is shorter, but there is a significant overshoot. The settling time grows but in this case it provides zero static error.
- When $T_d$ is increased, the rise time changes very little but the overshoot is reduced. The settling time is better and it has no influence in the steady state error.

As the steady-state error is already zero, the integral term is no needed. The proportional term and the derivate term are reduced and increased respectively so that the overshoot is lower. The selected constants and the response are given below:

$$K_{pz}=8$$
$$T_{iz}=\infty$$
$$T_{dz}=0.5$$

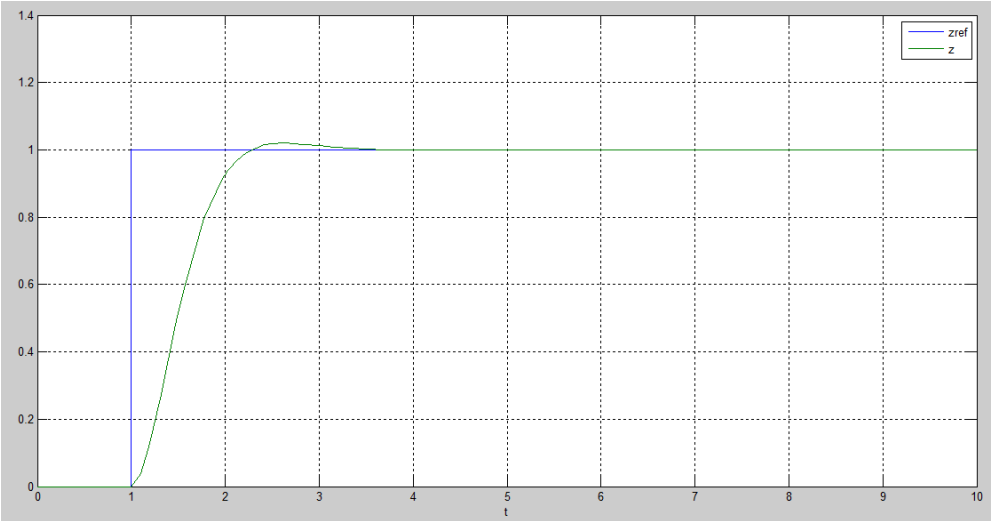



Figure 3.1-8: Tuned step response in z

As it can be seen, the response is so much better than the response in figure 3-10. The maximum overshoot in the z controller is a 2%, the rise time is 0.77 seconds and the settling time is 1.17 seconds.

The same procedure is done for the angles measured in degrees, when a unit step is introduced in each one of the commands, the result is:

| | $\phi$ | θ | $\psi$ | z |
|---|---|---|---|---|
| Kp | 4 | 2 | 1,5 | 8 |
| Ti | ∞ | ∞ | ∞ | ∞ |
| Td | 0,1833 | 0,275 | 0,42 | 0,5 |

Table 4: PID's selected constants

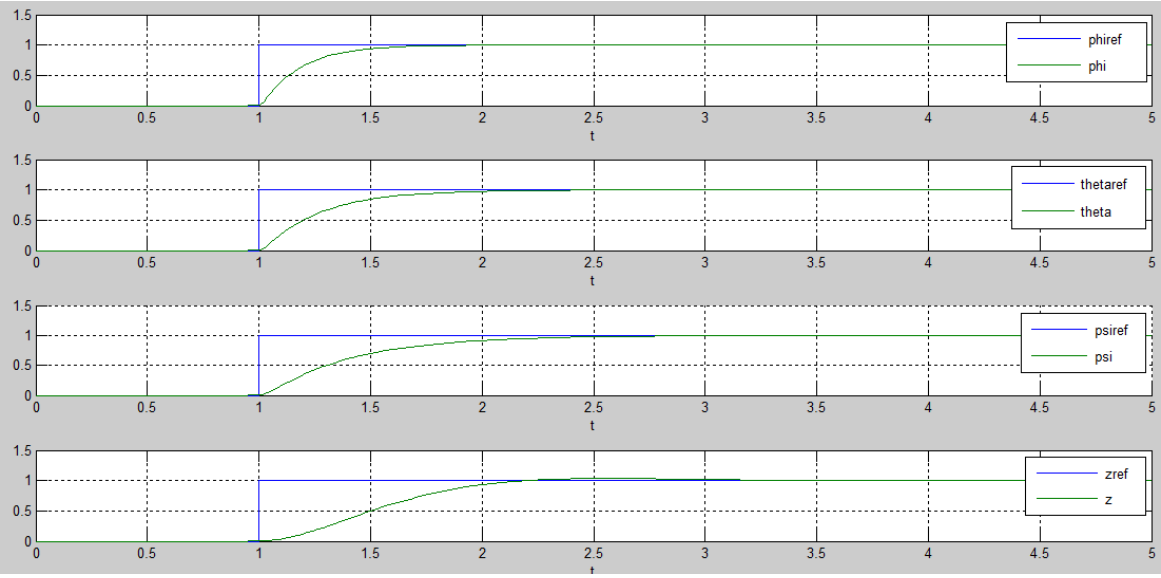



Figure 3.1-9: Step response for four PID controllers

The obtained results seem satisfactory but they might be optimized even more. Moreover, in the real quadcopter, the rotor's response is almost instantaneous. Since it is a low inertia rotor, starting and stopping are very fast which means that the changes of the attitude will probably be faster than the ones represented. Once these speeds are known, it will probably be needed to retune the PIDs.

Now, it will be checked if the controller works for other types of entries. In Figure 3.1-10 the entry commands and the responses are represented.

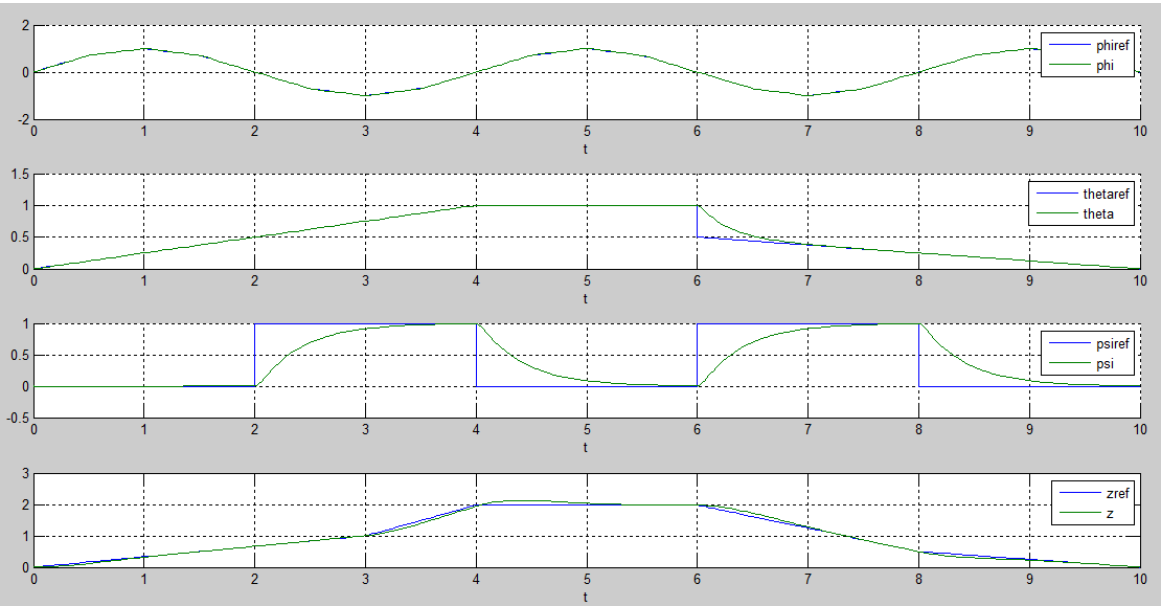



Figure 3.1-10: PID response for different entries

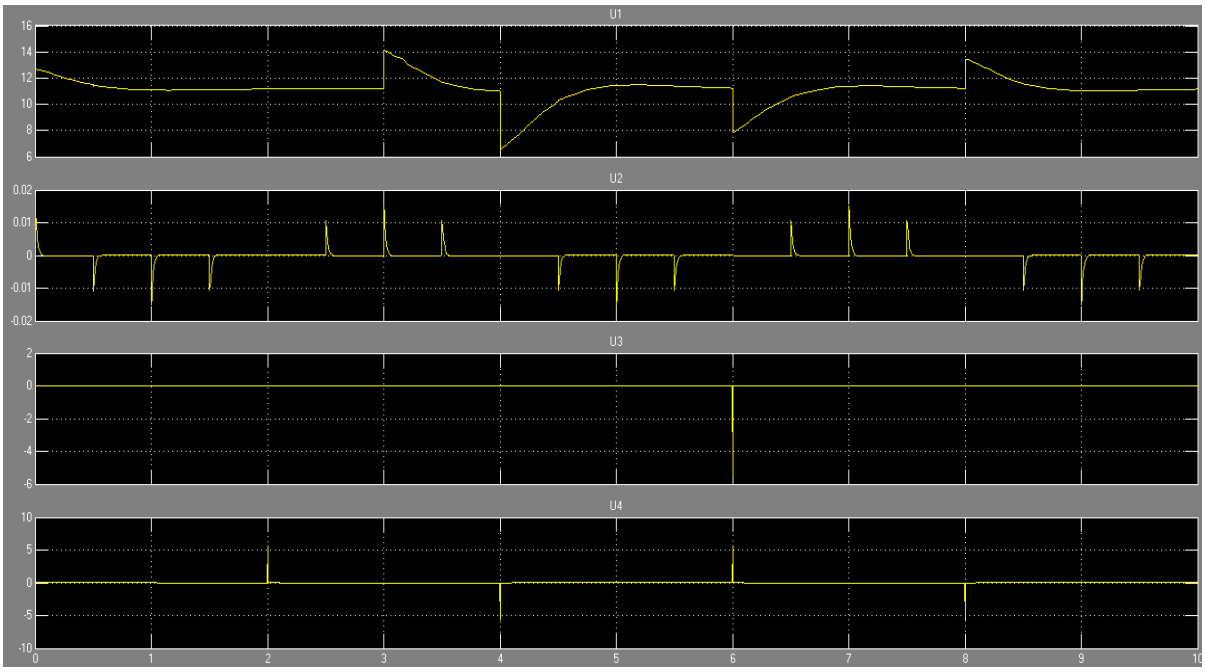



Figure 3.1-11: Thrust and torques for different entries

Ramp or sinusoidal signals are perfectly reproduced using the PID. In sudden ups and downs, the controller acts correctly but it takes more time to get the desired attitude. In Figure 3.1-11, the thrust ($U_1$) and the different torques are represented. It can be noticed that the bigger torques correspond to the sharp commands, while when the command is smooth, the needed torques are much smaller.

Some random noise will be included in the system of Figure 3.1-2. The objective is to know if the quadcopter can return to its desired commands (attitude and altitude) after some perturbations. In a real flight of a quadcopter the environmental disturbances such as wind gust and the ever-present sensor noise can greatly affect the position of the quadcopter.

First, a random noise will be introduced in the four parameters that are controlled. This noise is added to the feedback loop, so that, it could simulate that the measurements given by the instruments are not exactly the real attitude and altitude. In the following pictures the same entries of Figure 3.1-10 are represented for different amplitude of noise. In dark blue, the reference command is represented; in green, the output without noise and the following values have some noise added whose maximum amplitude is written in the graphs.

As it can be seen in Figure 3.1-12, the shape of the curve, especially when it is smooth (as in the first case), is maintained. However, as the noise is increased, the curve is sharper and resembles less to the original curve. Although the difference might not seem very big, it must be taken into account that a small difference in the desired angle can have an important influence in the position.

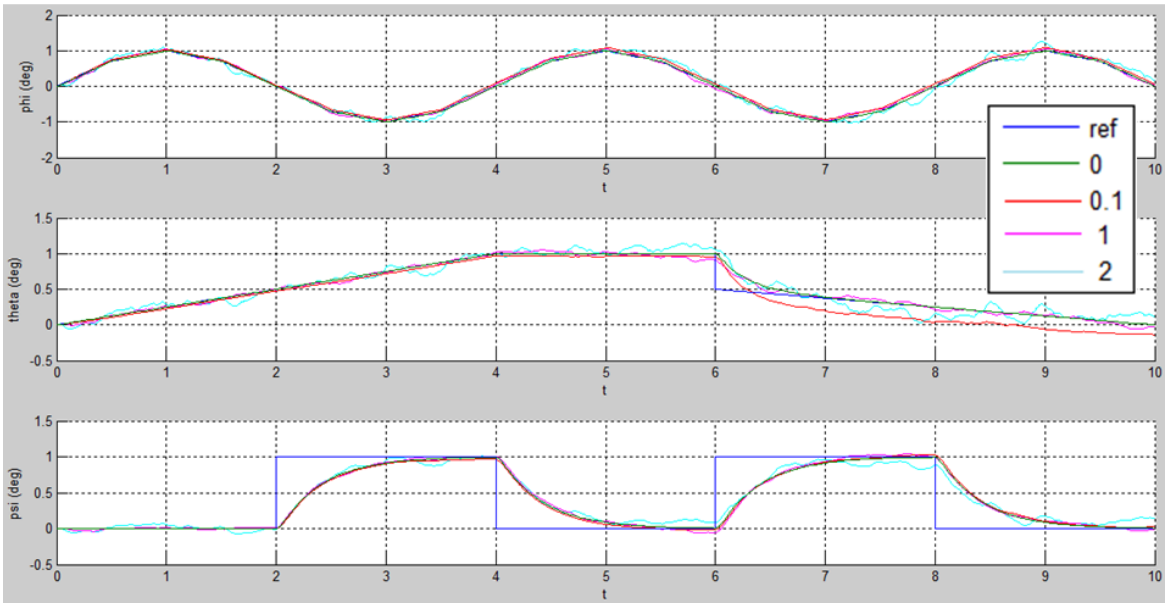



Figure 3.1-12: Attitude responses with random noise

Figure 3.1-13 represents the altitude output when there is some noise. The addiction of a random noise with maximum amplitude of 10 centimeters has already an important influence in the position and if the noise is increased, the final sign has a huge error.

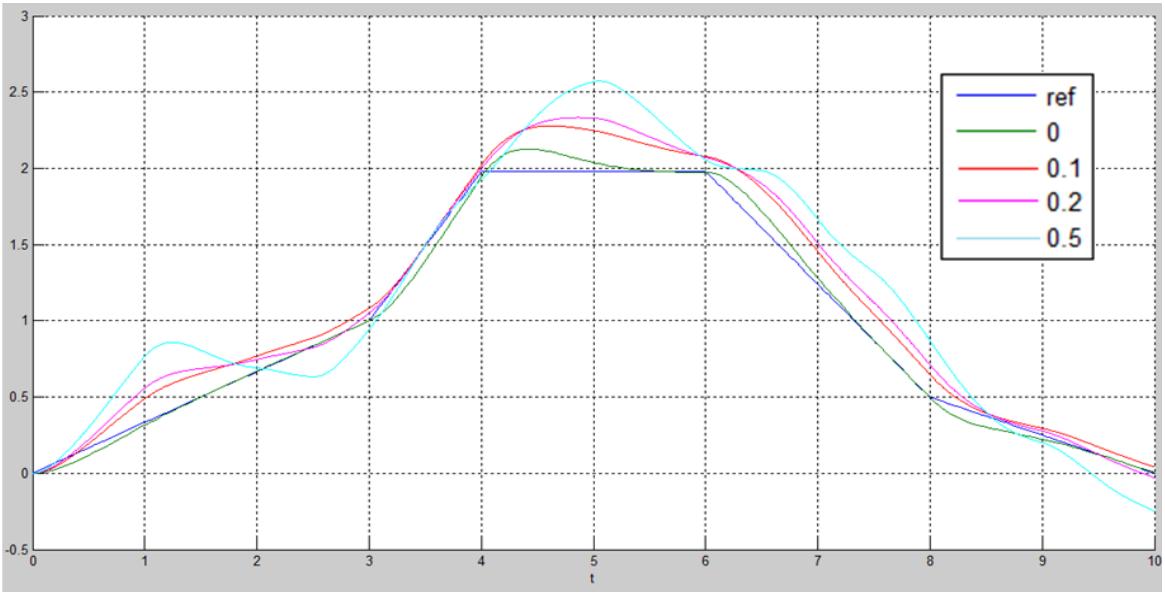



Figure 3.1-13: Altitude responses with random noise in meters

Secondly, some impulses are introduced at different instants of time, represented wind gusts to see how the system reacts. This disturbance has been introduced as pulse signals with duration of 0.5 seconds. In the angles phi, theta and psi a 3 degrees perturbation is introduced

at second 5, 7 and 9 respectively. The perturbation in the altitude is introduced at the third second and has a magnitude of one meter.

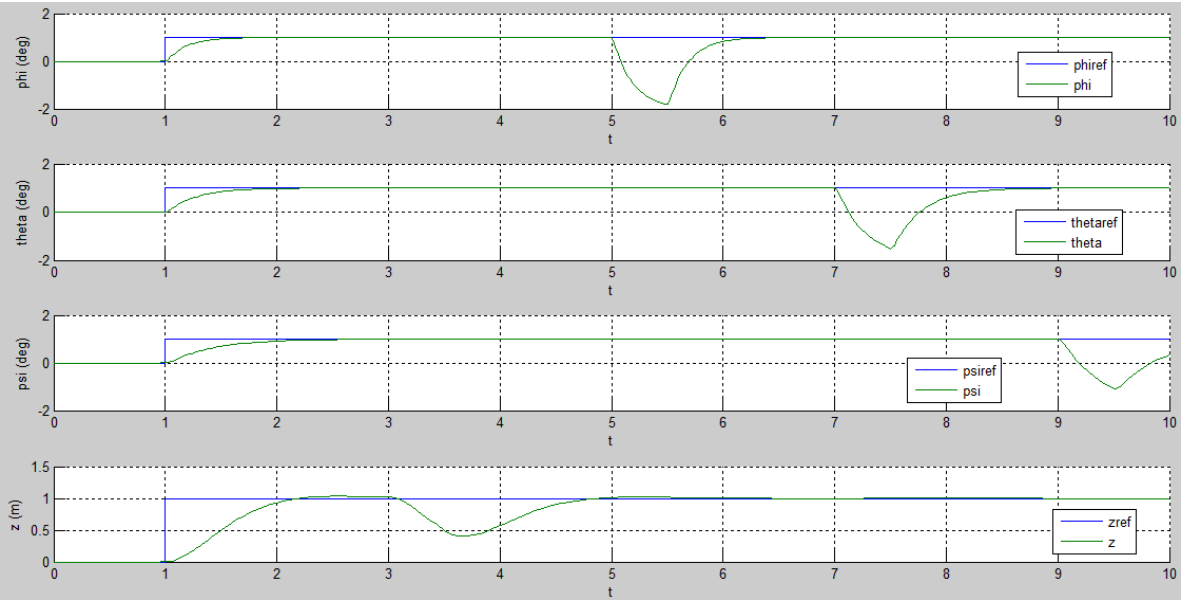



Figure 3.1-14: System responses with external disturbances

It can be clearly observed that, the vehicle leaves the attitude reference when a disturbance is introduced but then it reaches the reference again, obtaining a null steady-state error.

## 3.2 Control using Integral Backstepping

Using Integral Backstepping technique, a control law that forces the system to follow the reference trajectory can be designed. It is a recursive control methodology that makes use of Lyapunov stability theory. The algorithms in which the follow Simulink model will be based come from (BOUABDAHALL, 2007). The backstepping technique can be applied to nonlinear systems and has a strong robustness to disturbances. The quadrotor can be subjected to external actions, disturbances such as wind gusts or wind continued, that is the reason why the integral term is introduced. The benefit within the use of the integral term is to improve the system's steady state errors in closed-loop due to these perturbations, compared with simple backstepping approach.

This technique is designed for nonlinear systems that can be divided into n subsystems. Each one depends on itself and on the above and below subsystems except the nth subsystem that also depends on the control action of the nonlinear system. It has a recursive structure that controls gradually beginning for the first subsystem and adding a new subsystem control at each step. The process continues until the n subsystems are controlled and therefore the nonlinear system.

The control structure is detailed below:

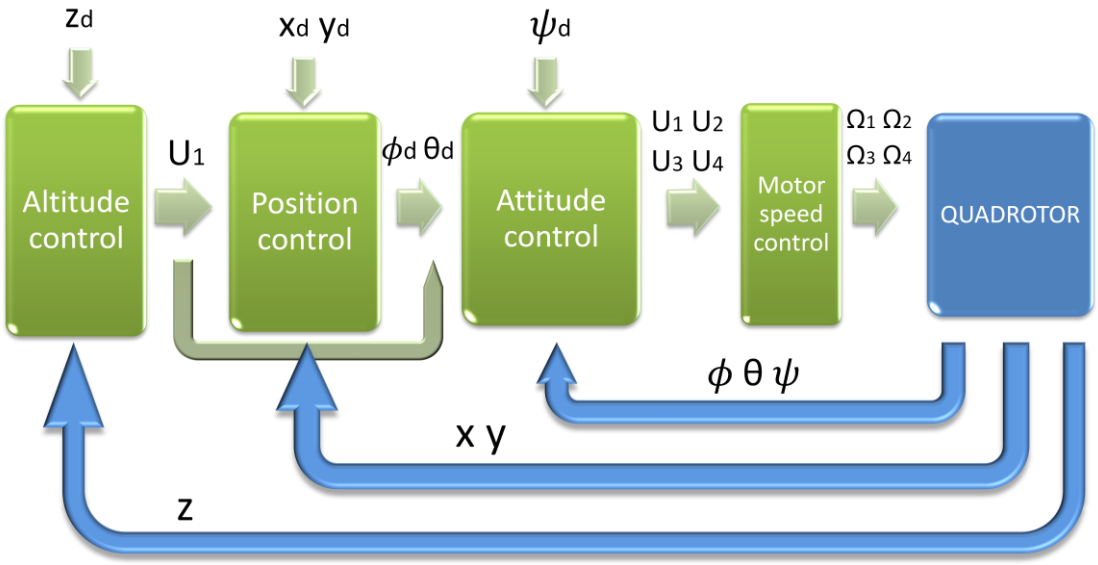



Figure 3.2-1: Diagram of the Integral Backstepping controller

In this picture, the values $x_d$, $y_d$, $z_d$ and $\psi_d$ are introduced by the user at each instant of time. The blue arrows represent the quadrotor's feedback. These values are obtained by different measurement instruments placed on it. The controller consists of three parts: attitude controller, altitude controller and position controller. Altitude controller outputs $U_1$ according to the desired altitude, the current altitude and the attitude angles. The second controller receives the calculated thrust and combines the desired position with the current position to obtain the desired roll and pitch angles. Finally, the attitude controller receives the desired angles and the current ones to output $U_2$, $U_3$ and $U_4$. These values are introduced in the dynamic model of the quadcopter.

The notation that will be used in this section in order to condense the equations is:

$$a_1 = \frac{I_y - I_z}{I_x} \qquad a_2 = \frac{J_r}{I_x} \qquad b_1 = \frac{1}{I_x}$$

$$a_3 = \frac{I_z - I_x}{I_y} \qquad a_4 = \frac{J_r}{I_y} \qquad b_2 = \frac{1}{I_y} \tag{76}$$

$$a_5 = \frac{I_x - I_y}{I_z} \qquad b_3 = \frac{1}{I_z}$$

$$u_x = \cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi \tag{77}$$

$$u_y = \cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi$$

$$\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{78}$$

The equations of motion are now written as:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}a_1 + \dot{\theta}\,a_2\Omega_r + b_1U_2 \qquad \ddot{X} = \frac{U_1}{m}u_x$$

$$\ddot{\theta} = \dot{\theta}\dot{\psi}a_3 + \dot{\phi}a_4\Omega_r + b_2U_3 \qquad \ddot{Y} = \frac{U_1}{m}u_y \tag{79}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}a_5 + b_3U_4 \qquad \ddot{Z} = \frac{U_1}{m}\cdot(\cos\phi\, cos) - g$$

### *3.2.1 Attitude Control*

The objective of this control system is to maintain the quadcopter with the proper orientation. Three separate controllers are designed to track the desired roll, pitch, yaw angles. The last one is introduced directly by the user; the other two are the outputs of another controller. The derivation is similar for the three angles. For this reason, only the calculations of the roll angle will be developed. The non-linear system that has been already seen was:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}a_1 + \dot{\theta}\,a_2\Omega_r + b_1U_2 \tag{80}$$

The law control that will be implemented needs to be stable. To be sure of that, a Lyapunov function V(x,u) is introduced. This function is continuous, defined positive (except in x=0 where it is zero) and for each state x, a control u that will brings the system to zero can be found.

The first step is to define the tracking error of the system, which is the difference between the desired roll angle and the actual angle. The actual angle comes from the feedback of the system and it is obtained from the gyroscopes.

$$e_1 = \phi_d - \phi \tag{81}$$

The purpose of this change of variable is getting no error in steady state, which means: $lim_{t\to\infty} e_1(t) = 0$. Its derivative with respect to time is:

$$\frac{de_1}{dt} = \dot{\phi}_d - \omega_x \tag{82}$$

The value $\omega_x$ is not a control input. It is given a desired behavior so that it is consider as a virtual control.

$$\omega_{xd} = c_1e_1 + \dot{\phi}_d + \lambda_1\chi_1 \tag{83}$$

Where $\chi_1 = \int_0^t e_1(\tau)d_\tau$ and $c_1, \lambda_1$ are positive constants.

The last equation (83) is solved for $\dot{\phi}_d$ and introduced in the equation (82).

$$\frac{de_1}{dt} = -c_1 e_1 - \lambda_1 \chi_1 + \omega_{xd} - \omega_x \tag{84}$$

Otherwise, the virtual control that has been created is also going to have an error with respect to the real $\omega_x$.

$$e_2 = \omega_{xd} - \omega_x \tag{85}$$

This way, the equation (84) can be expressed as:

$$\frac{de_1}{dt} = -c_1 e_1 - \lambda_1 \chi_1 + e_2 \tag{86}$$

Replacing the value $\omega_{xd}$ by the equations (83) and (85), $e_2$ is expressed as:

$$e_2 = c_1 e_1 + \dot{\phi}_d + \lambda_1 \chi_1 - \omega_x \tag{87}$$

This equation is derivative with respect to time obtaining the following result:

$$\frac{de_2}{dt} = c_1 \cdot \frac{de_1}{dt} + \ddot{\phi}_d + \lambda_1 e_1 - \ddot{\phi} \tag{88}$$

The equations (86) and the $\ddot{\phi}$ part of the equation (79) are introduced in the formula above:

$$\frac{de_2}{dt} = c_1 \cdot (-c_1 e_1 - \lambda_1 \chi_1 + e_2) + \ddot{\phi}_d + \lambda_1 e_1 - \dot{\theta}\dot{\psi} a_1 - \dot{\theta}\, a_2 \Omega_r - b_1 U_2 \tag{89}$$

The selected Lyapunov function is:

$$V = \frac{1}{2}\lambda({\lambda_1 \chi_1}^2 + {e_1}^2 + {e_2}^2) \tag{90}$$

This equation is derivate and the value $\dot{e_1}$ is substitute by the equation (86). The desirable value for the time derivation of $e_2$, which makes $\dot{V} < 0$ for every value except 0, is given below where $c_2$ is a positive constant. This means that the function is stable.

$$\frac{de_2}{dt} = -c_2 e_2 - e_1 \tag{91}$$

This value is placed in the equation (89) which is solved for $U_2$:

$$U_2 = \frac{1}{b_1}\left[(1 - c_1^2 + \lambda_1)e_1 + (c_1 + c_2)e_2 - c_1 \lambda_1 \chi_1 + \ddot{\phi}_d - \dot{\theta}\dot{\psi} a_1 + \dot{\theta} a_2 \Omega_r\right] \tag{92}$$

The process is similar for the other two attitude controls, for pitch and yaw it is obtained:

$$U_3 = \frac{1}{b_2}\left[(1 - c_3^2 + \lambda_2)e_3 + (c_3 + c_4)e_4 - c_3\lambda_2\chi_2 + \ddot{\theta}_d - \dot{\phi}\dot{\psi}a_3 - \dot{\phi}a_4\Omega_r\right] \quad (93)$$

$$U_4 = \frac{1}{b_3}\left[(1 - c_5^2 + \lambda_3)e_5 + (c_5 + c_6)e_6 - c_5\lambda_3\chi_3 + \ddot{\psi}_d - \dot{\phi}\dot{\theta}a_5\right] \quad (94)$$

with $c_3, c_4, c_5, c_6, \lambda_2, \lambda_3 > 0$.

### 3.2.2 Altitude Control

This control system allows the quadcopter to fly at the desirable altitude. The system inputs the desirable altitude, the feedback of the actual altitude that comes from the measures taken by the barometer and also the feedback of the angles. It outputs the thrust that will be introduced in the position control system. The process is the same as in the attitude control; the only different is that now the equation of motion is:

$$\ddot{Z} = \frac{U_1}{m} \cdot (\cos\phi\, cos) - g \quad (95)$$

The errors are express as:

$$e_7 = z_d - z \qquad e_8 = c_7 e_7 + \dot{z}_d + \lambda_4\chi_4 - \dot{z} \quad (96)$$

The altitude control law is:

$$U_1 = \frac{m}{cos\phi cos\theta}[g + (1 - c_7^2 + \lambda_4)e_7 + (c_7 + c_8)e_8 - c_7\lambda_4\chi_4 + \ddot{z}_d] \quad (97)$$

with $c_7, c_8, \lambda_4, \lambda_3 > 0$.

### 3.2.3 Position Control

The position control has the objective of maintaining the quadcopter at the desired point in each instant of time. This purpose is achieved by changing the quadcopter's roll and pitch angles. Therefore, the inputs of the system are the desired positions in x and y, the position feedback obtained by the IMU (Inertial Measurement Unit) as well as the thrust. The outputs are the desired angles that will be introduced in the attitude control system. For small angles, the constants $u_x$ and $u_y$ become:

$$u_x = \cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi = \theta \quad (98)$$

$$u_y = \cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi = -\phi \quad (99)$$

The equations of motion in x and y are:

$$\ddot{X} = \frac{U_1}{m}\theta \qquad \ddot{Y} = -\frac{U_1}{m}\phi \tag{100}$$

and the tracking errors:

$$e_9 = x_d - x \qquad e_{10} = c_9 e_9 + \dot{x}_d + \lambda_5 \chi_5 - \dot{x}$$
$$e_{11} = y_d - y \qquad e_{12} = c_{11} e_{11} + \dot{y}_d + \lambda_6 \chi_6 - \dot{y} \tag{101}$$

After following the same procedure as in section 3.2.1, it is obtained:

$$\theta_d = \frac{m}{U_1}[(1 - c_9^2 + \lambda_5)e_9 + (c_9 + c_{10})e_{10} - c_9\lambda_5\chi_5] \tag{102}$$

$$\phi_d = -\frac{m}{U_1}[(1 - c_{11}^2 + \lambda_6)e_{11} + (c_{11} + c_{12})e_{12} - c_{10}\lambda_6\chi_6] \tag{103}$$

with $c_9, c_{10}, c_{11}, c_{12}, \lambda_5, \lambda_6 > 0$.

### *3.2.4 Implementation in Simulink*

This second controller has also been built in Simulink. The entries of the whole system are the desired position and yaw angle as it has been said. The model in Simulink has several subsystems. The quadcopter and the motor speed control blocks are the same as in the previous model. The controller is composed of four blocks represented in yellow in Figure 3.2-2. Moreover, the current position of the quadcopter as well as its attitude is feedbacked to the different blocks of the controller.

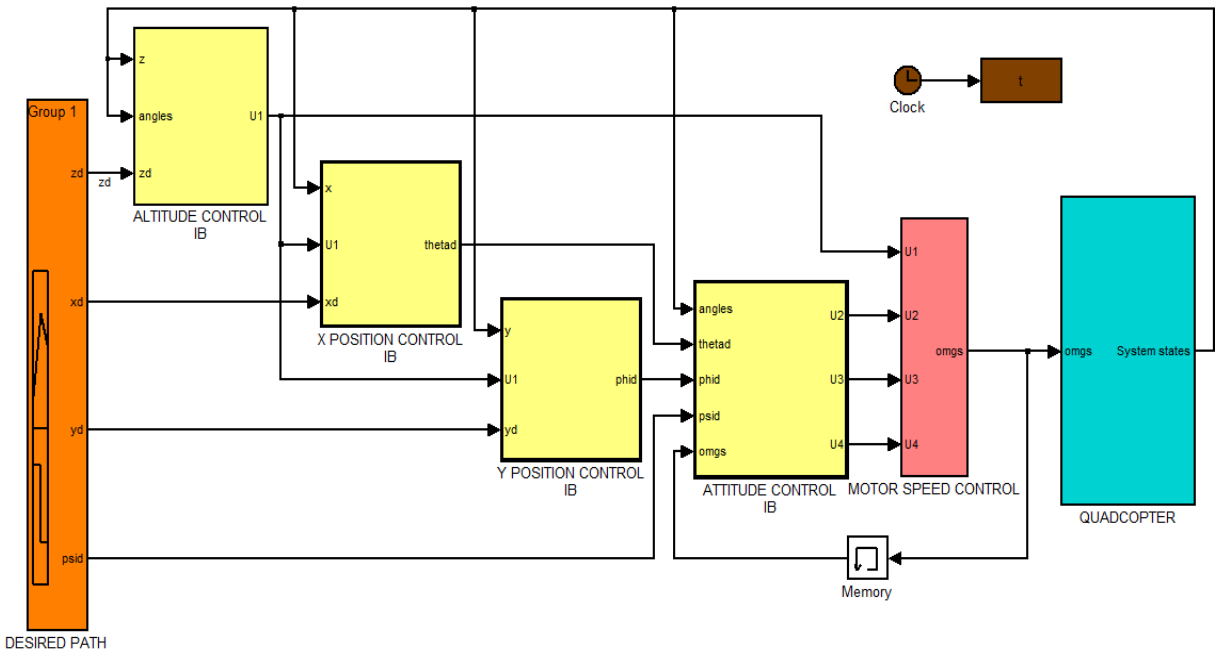



Figure 3.2-2 IB controller in Simulink

When designing the Simulink model, one of the key constraints is to avoid algebraic loops. Basically, algebraic loops occur when the input of a block is the output of the same block, the value of the input directly controls the value of the output. Algebraic loops are difficult to solve mathematically, this slows down the simulation and might even cause the simulation to stop. Simulink have an algebraic loop solver that attempts to solve the situation in an iterative way and in a lot of cases it succeeds. However, it is better to try to avoid this blocks whenever it is possible. By adding a small delay the algebraic loop can be eliminated.

As it can be seen, the equations (92) and (93) contain the term $\Omega_r$ which means that the propellers speed need to be introduced in the attitude controller. The propeller angular rates come directly from the motor speed control block and generate an algebraic loop. In this case, the problem has been solved by adding a memory block. The Memory block outputs its input from the previous time step, applying a one integration step sample-and-hold to its input signal. However, when adding small delays, it must always be ensured that unnecessary dynamics are not being added because they may affect the results.

Each one of the yellow blocks contains one of the equations calculated in the previous section, or three in the case of the attitude controller. In the following fFigure 3.2-3, the Simulink model for the altitude controller is represented. For the other controllers the model is very similar.

$$U_1 = \frac{m}{cos\phi cos\theta}[g + (1 - c_7^2 + \lambda_4)e_7 + (c_7 + c_8)e_8 - c_7\lambda_4\chi_4 + \ddot{z}_d]$$

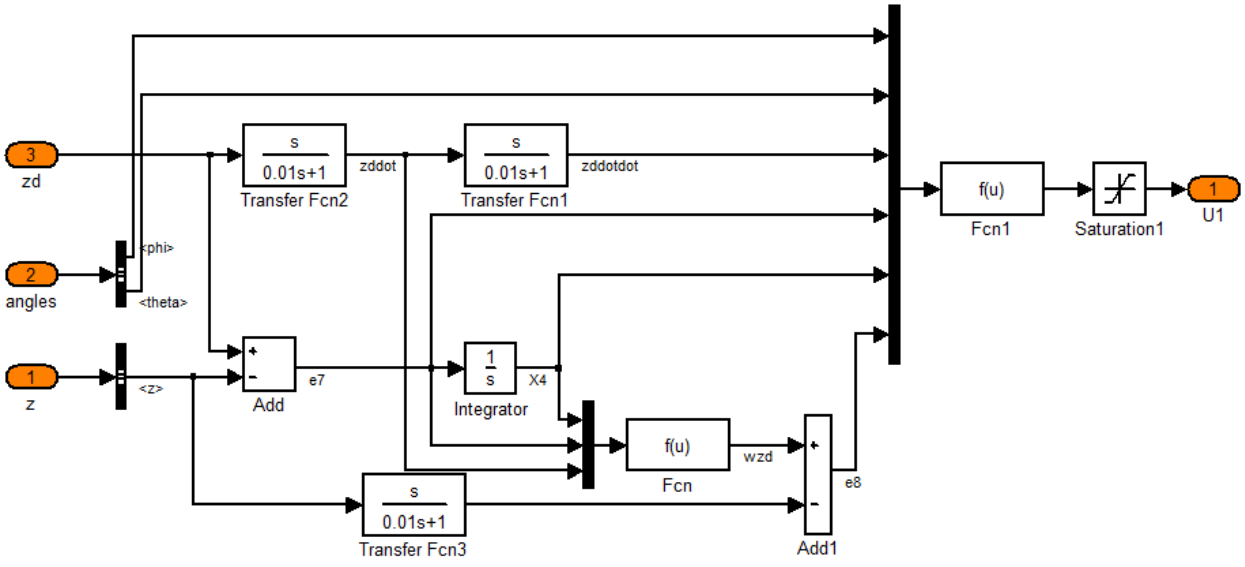



Figure 3.2-3 IB Altitude control block

The equations including in the functions blocks are defined in the following equations:

Block Fcn:

$$f(u) = c7 * u(2) + u(3) + ld4 * u(1)$$

Block Fcn1:

$$f(u) = m * (9.81 + (1 - (c7\text{^}2) + ld4) * u(4) + (c7 + c8) * u(6) - c7 * ld4 * u(5) + u(3))/(cos(u(1)) * cos(u(2)))$$

where the term u(i) represents the entry of the system in the order that they are introduced in the Mux block.

At the beginning, all the derivations blocks of the system had been introduced as derivative blocks like the one shown in fFigure 3.2-4.

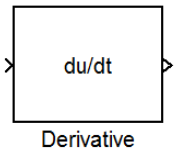
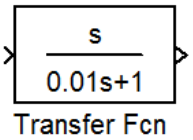



Figure 3.2-4 Derivative block

This block outputs the derivation of the entry. The accuracy of the results depends on the size of the steps taken in the simulation. The more steps, the more smoother and accurate is the output curve. However, when the input is a discrete signal or has an instant with infinite slope the output of the system is an impulse which greatly alters the results of the whole system, obtaining odd global outputs. To solve this problem the block was substituted by a transfer function block represented below:




In this manner, the solution is linearized. By this action, the block is approximated to the transfer function of a high pass filter. A pole has been added in s=100. In the following pictures the output of this block is compared with the derivate one:

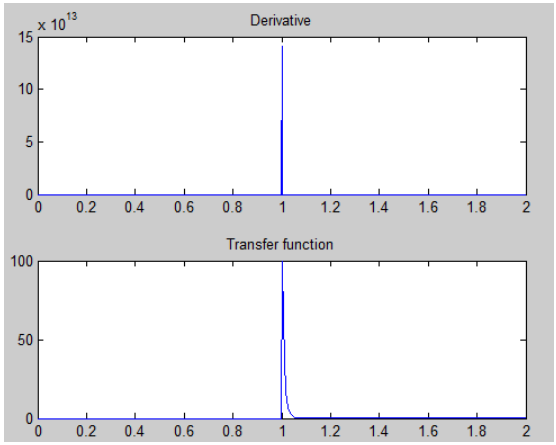



Figure 3.2-5 Comparison for a unit step entry in t=1

As it can be seen in fFigure 3.2-5 the block has reduced the magnitude of the impulse with a zero response for the rest of the instants. The problem of the system going to the infinite is solved. Some other entries are tested to see if the block behaviors as the derivate one.

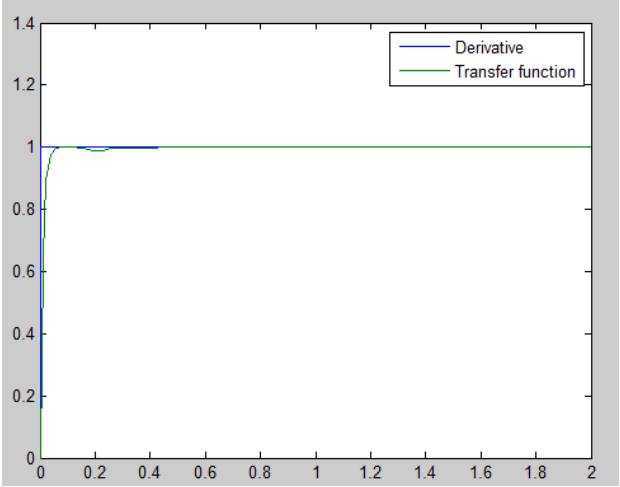



Figure 3.2-6 Comparison for a ramp entry of slope 1

For a ramp entry, the error in the instant 0.05 is already smaller than the 1% being almost zero ($10^{-9}$) from there.

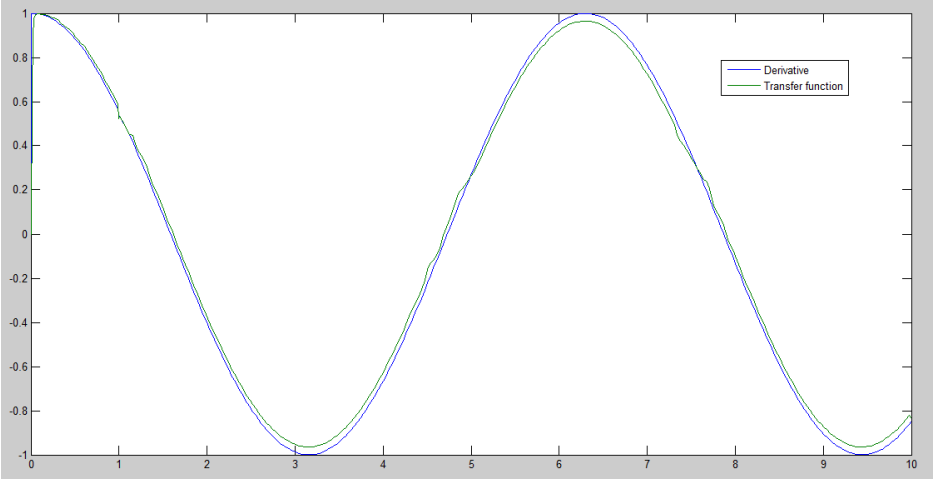



Figure 3.2-7 Comparison for a sine entry

The response with the transfer function is more attenuated but it reproduces exactly the shape of the real response. Due to these results, the transfer function is validated to be used instead of the derivation block.

### *3.2.5 Simulation results*

The proposed control strategy has been tested by simulation in order to check its performance. As it has been seen in the previous section, some constants must be adjusted to obtain a path that resembles as much as possible to the desired path.

Three constants are selected for each one of the six degrees of freedom. The behavior of these three constants can be likened to the case of PID controllers. For example in the case of the phi attitude controller, $c_1$ is proportional to the tracking error, like $K_p$ in the PID. $c_2$ and $\lambda_1$ are proportional to the derivation and integration of the error, as $K_d$ and $K_i$ in the PID controller.

To see how the value of the constants affects the result, a step has been introduced in the controller IB and the constants $c_1$, $c_2$ and $\lambda_1$ have been modified.

In the following picture, the output of a step input is represented for different values of $c_1$ when the other $c_2$ and $\lambda_1$ are held constant. It can be observed that this value greatly affects the overshoot, the stability and the settling time.

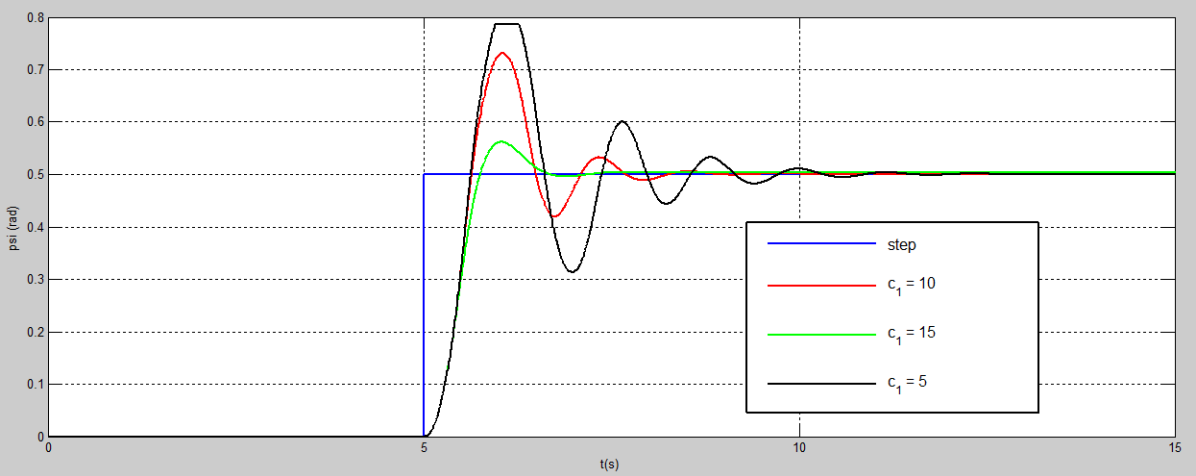



Figure 3.2-8 IB output for different values of $c_1$

The output is represented for different values of $c_2$ when the other $c_1$ and $\lambda_1$ are held constant. The constants are modified depending on these values; however, the changes are lighter than in the previous case.

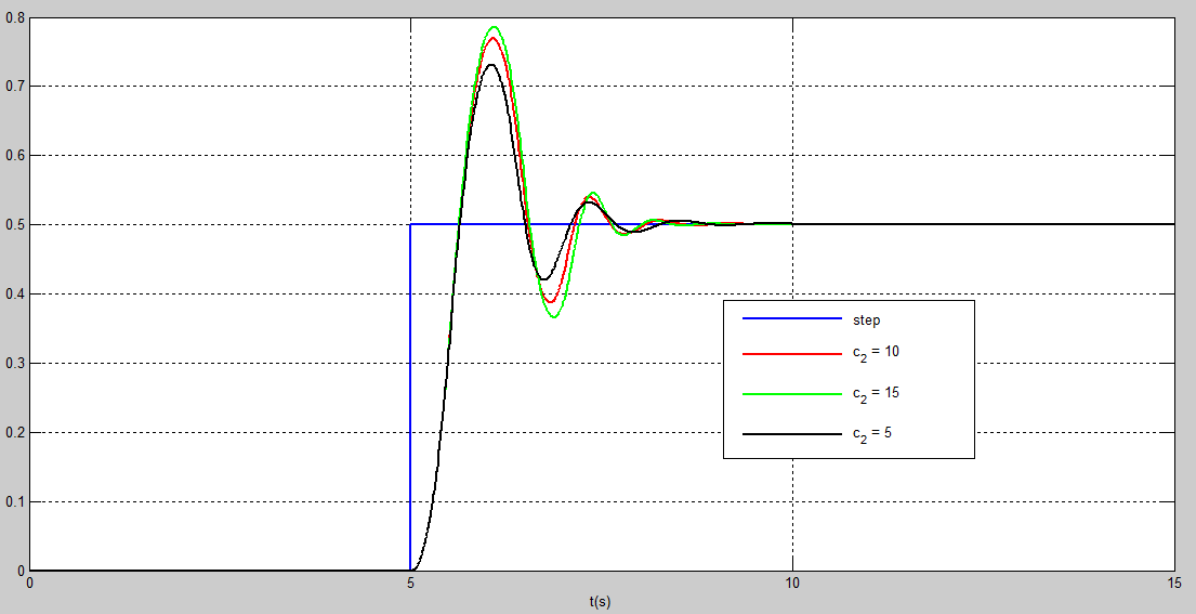



Figure 3.2-9 IB output for different values of $c_2$

The last parameter is modified in Figure 3.2-10. The values change much less than in the other two cases, however, the steady state error is corrected.

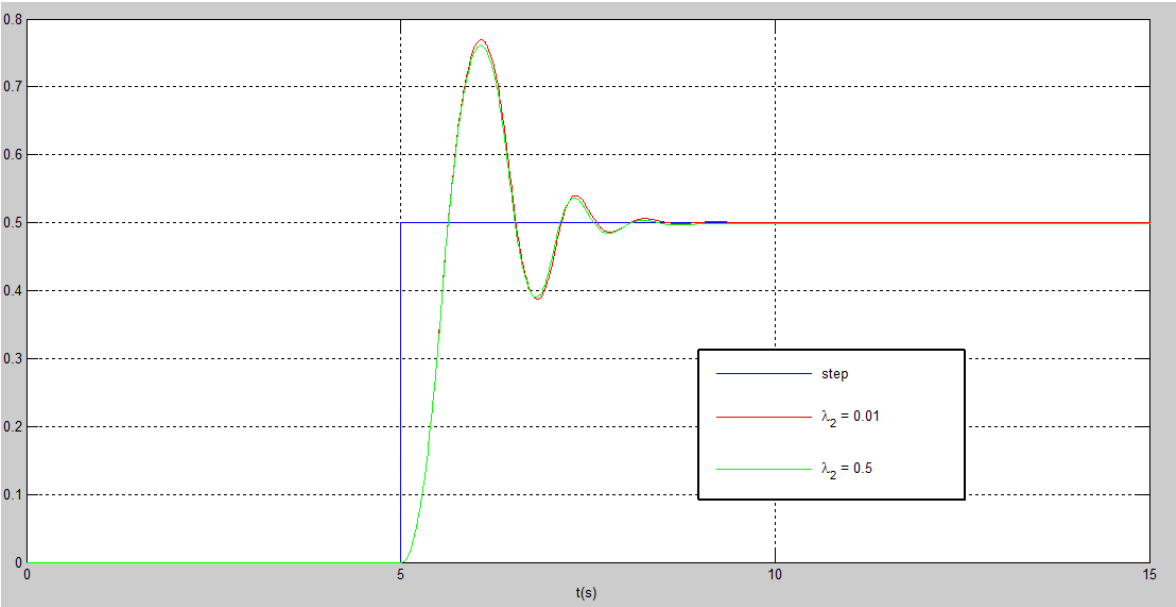



Figure 3.2-10 IB output for different values of λ

As it was done with the PID, the following chart represents the variation of the output characteristics when the constants are increased:

| | Rise time | Overshoot | Settling time | Steady State error |
|---|---|---|---|---|
| $C_1$ | Increase | Decrease | Decrease | Increase |
| $C_2$ | Decrease | Increase | Increase | Decrease |
| $\lambda_1$ | Increase | Decrease | Decrease | Eliminate |

Table 5: Behavior of the IB constants

The reference path that must be followed by the quadcopter is an ascent in the z axis continuing with a square at constant height. Some variables for this case are:

| phi | | | theta | | | psi | | |
|---|---|---|---|---|---|---|---|---|
| c1 | c2 | λ1 | c3 | c4 | λ2 | c5 | c6 | λ3 |
| 15 | 6 | 0,07 | 15 | 6 | 0,05 | 10 | 5 | 0,01 |

| x | | | y | | | z | | |
|---|---|---|---|---|---|---|---|---|
| c9 | c10 | λ5 | c11 | c12 | λ6 | c7 | c8 | λ4 |
| 10 | 1,5 | 0,001 | 10 | 1,5 | 0,001 | 3,5 | 1,5 | 0,01 |

Table 6: IB's selected constants

In the following pictures, the reference trajectory and the actual trajectory of the vehicle are represented.

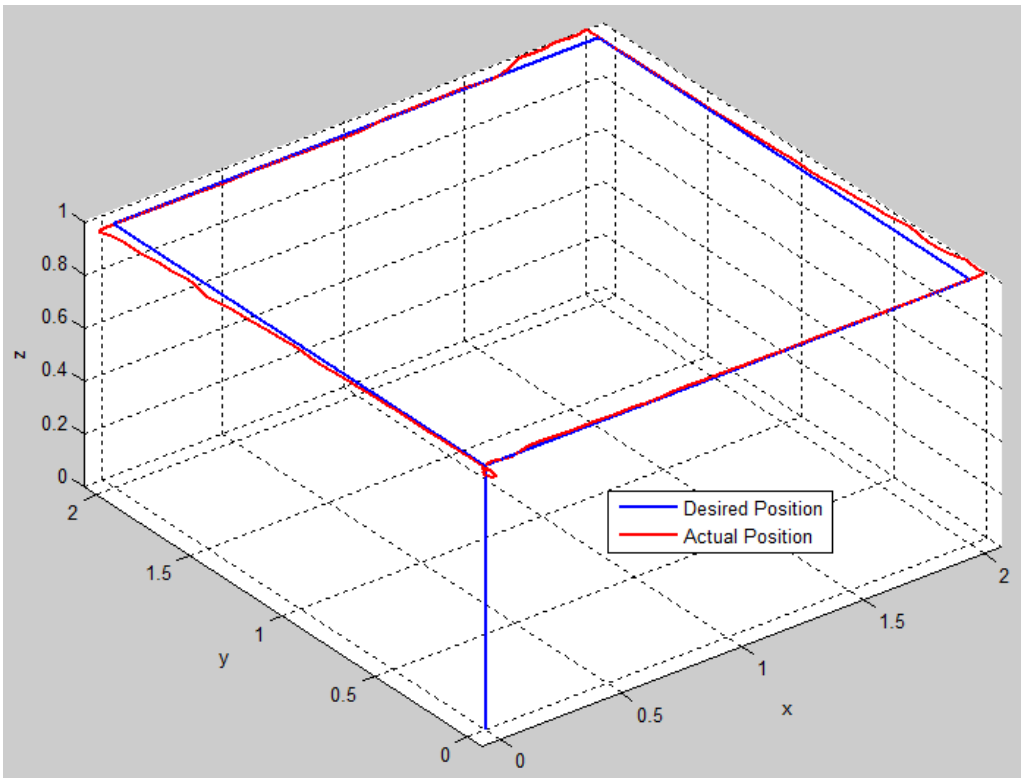



Figure 3.2-11 Desired and current position for IB controller in 3D

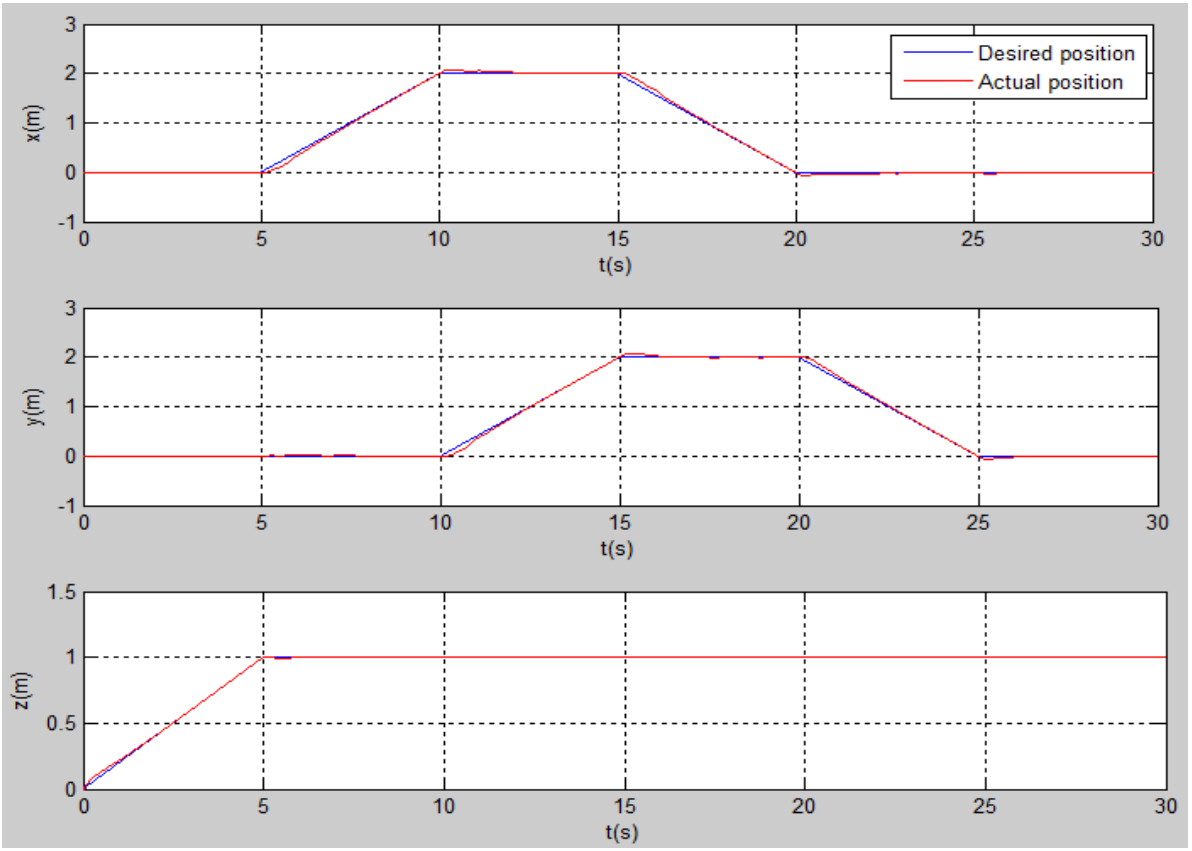



Figure 3.2-12 Desired and current position for IB controller in the three axes

The error between these two trajectories is represented below. As it is normal, the error increases when a slope change occurs. However after a short while the quadcopter is able to recover its desired path with zero steady state error in the three axes. It can be also observed that and the error produced by the change in slope in x and y, affects the z position.

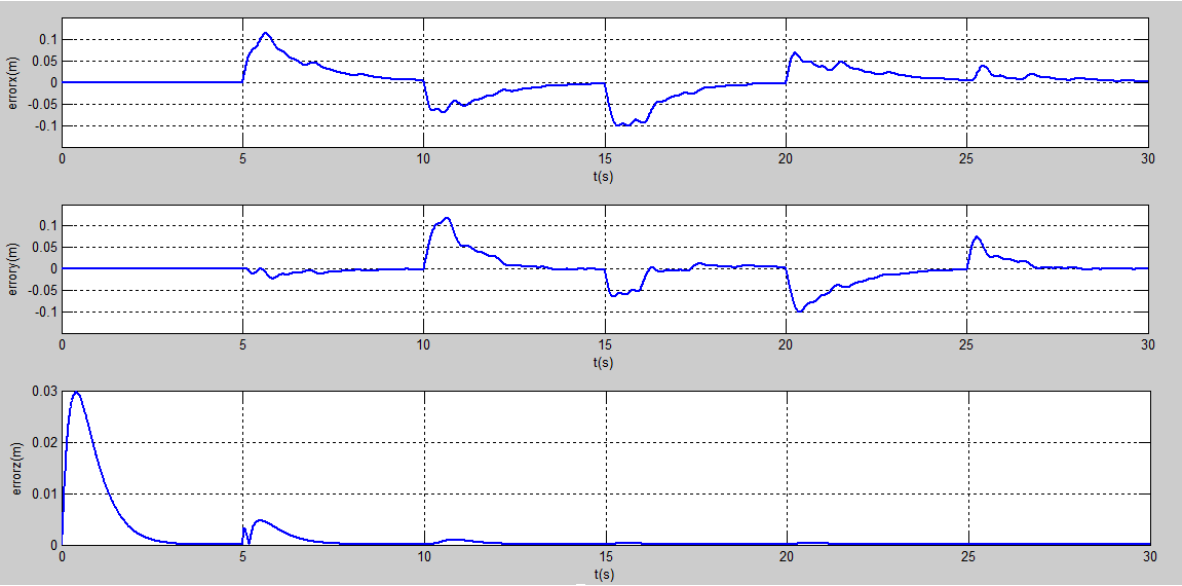



Figure 3.2-13 Trajectory tracking error with IB

The x and y position controllers output the desired angles theta and phi respectively. After that the attitude controller selects the proper torques to obtain those angles. To check if the attitude controller is tuned correctly, in Figure 3.2-14 the desired angles are compared with the current angles.

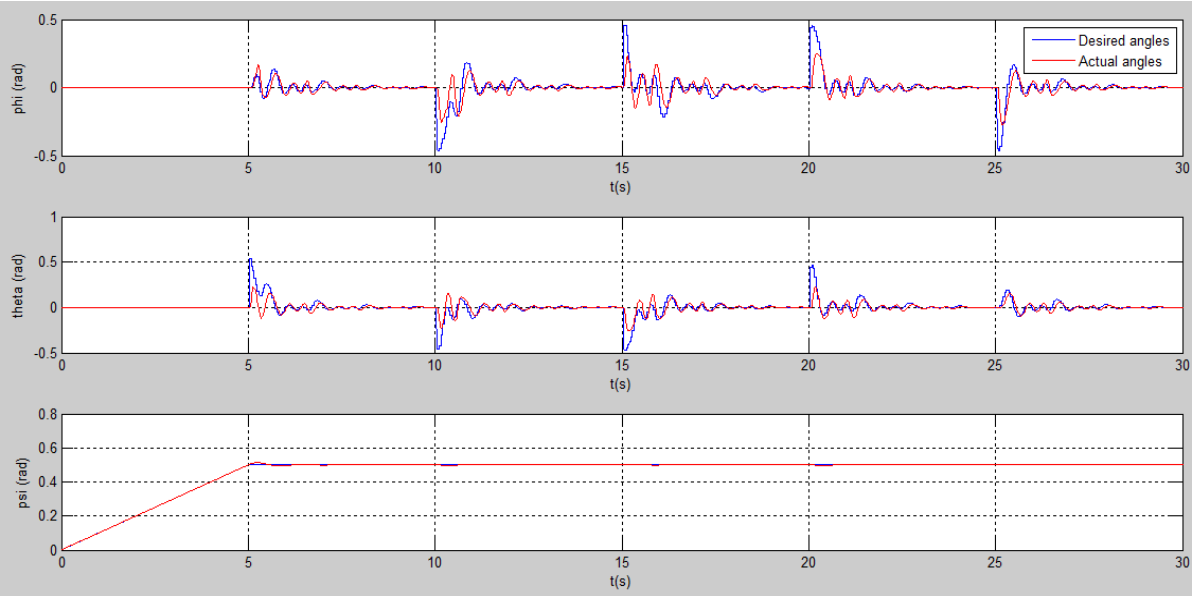



Figure 3.2-14 Desired and current attitude for IB controller

The angle psi, whose desired value is defined directly by the user, is followed perfectly. The desired angles selected by the controller (phi and theta) are bigger than the values that the quadcopter actually achieved. The quadcopter response is not as fast as it should be to obtain zero error. This is illustrated in Figure 3.2-12 the quadcopter remains at its last position longer than the ordered.

Figure 3.2-15 represents the total thrust and torques commands that are introduced into the quadcopter. In the blue graphic it can be observed at second five how much the quadcopter must reduce its thrust in order to reduce its speed in the z axis to zero. Each time the slope changes, $U_2$ and $U_3$ suffer a huge raise (or descent) and the saturation value in $U_2$ and $U_3$, obtained in 2.4.4 is usually achieved. It is very important to introduced saturation values in the model. In this case, if they are not introduced when the first change in the slope occurs, the torque goes to the infinity and then all the coordinates followed it.

One way to reduce this abrupt torques could be introducing the slope changes in a smoother form, as it will be discussed in section 3.4.

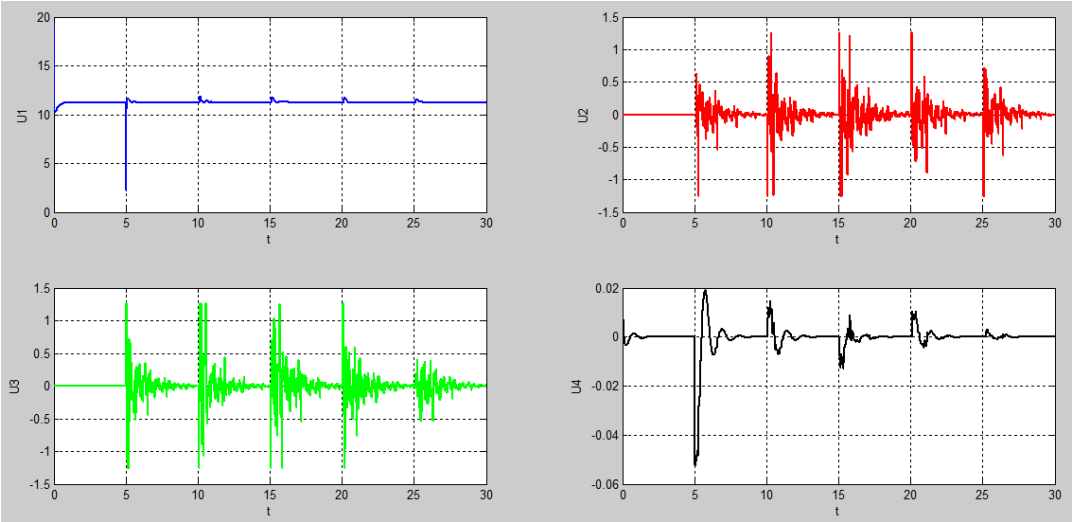



Figure 3.2-15 Total thrust and torques in the IB controller

A second reference path has been simulated. Some variables have been slightly modified to obtain a most accurate reproduction of the path. In this case is a helicoidally trajectory in the $R^3$ Cartesian space defined by:

$$x_r = 0.5 \cdot \cos t$$

$$y_r = 0.5 \cdot \sin t \qquad (104)$$

$$z_r = 0.5 \cdot \cos \frac{t}{10}$$

$$\psi = 0$$

It is important to define the initial constants properly, if not, the system can produce erroneous results or even not be able to simulate. The initial constants introduced are:

$$x_{0_r} = 0.5 \qquad y_{0_r} = 0 \qquad z_{0_r} = 0$$
$$\dot{x}_{0_r} = 0 \qquad \dot{y}_{0_r} = 0.5 \qquad \dot{z}_{0_r} = 0.05 \qquad (105)$$

The variables that have been selected are the same as in the previous case with the exception of c9 and c11 that have been increased from 10 to 15.

The following pictures show the simulation results of the path following of this reference trajectory. In the same figures, it has also been represented the quadcopter's path when some disturbances on the aerodynamic forces are introduced. These forces are introduced in the quadcopter model with a duration of half a second and a value of one Newton. The forces are reproduced in the following picture.

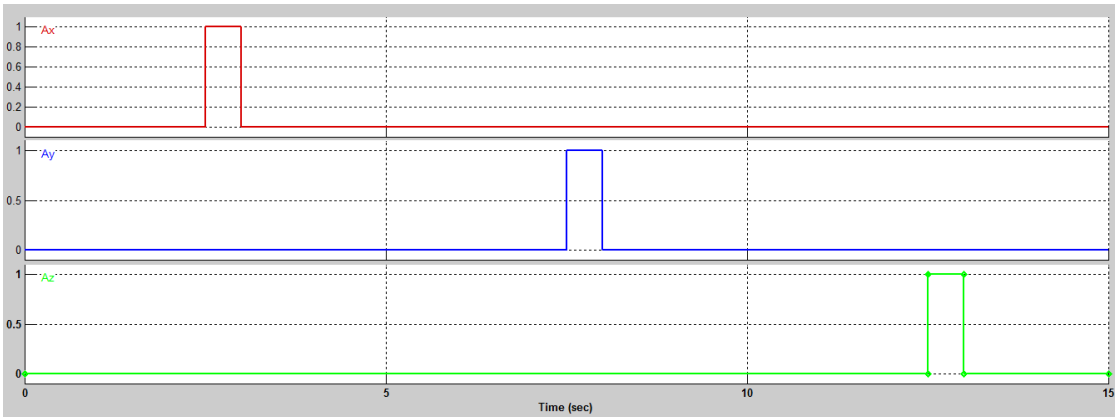



Figure 3.2-16 Perturbations introduced in the IB controller

In fFigure 3.2-17 and Figure 3.2-18 the reference positions of the quadcopter as well as its actual position with and without disturbances are represented. Some marks have been included indicating the moments in which the disturbances have been applied.

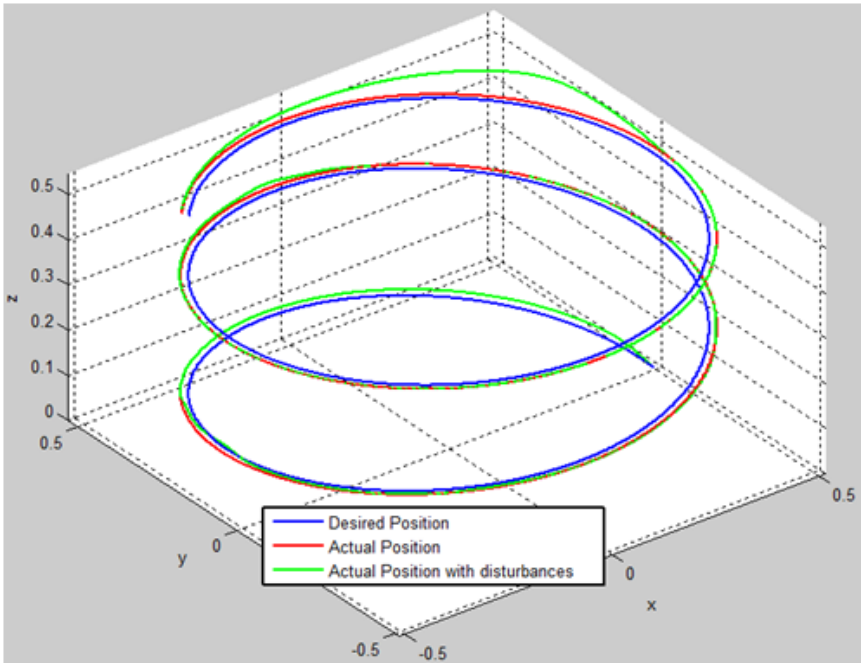



Figure 3.2-17 Desired and current position for an helicoidally entry

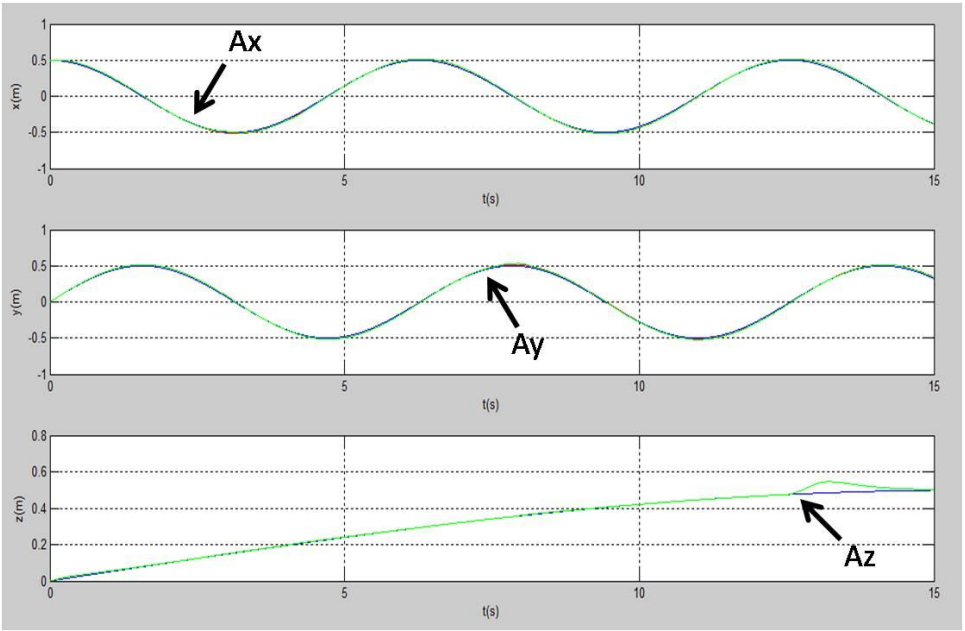



Figure 3.2-18 Desired and current position for an helicoidally entry in the three axes

In Figure 3.2-19 the error with respect to the reference path with no disturbances (red) and with disturbances (green) is represented. It can be observed that even in the previous pictures, only the error in z can be appreciated, in the other axis the error increases when the perturbation is introduced. However, the system recovers again its desired trajectory. The error in the z axis is the smallest one, of the order of $10^{-3}$, but is also the one that suffers most the effects of external disturbances. In the x and y axes, the maximum error is less than 2 inches when no disturbances.

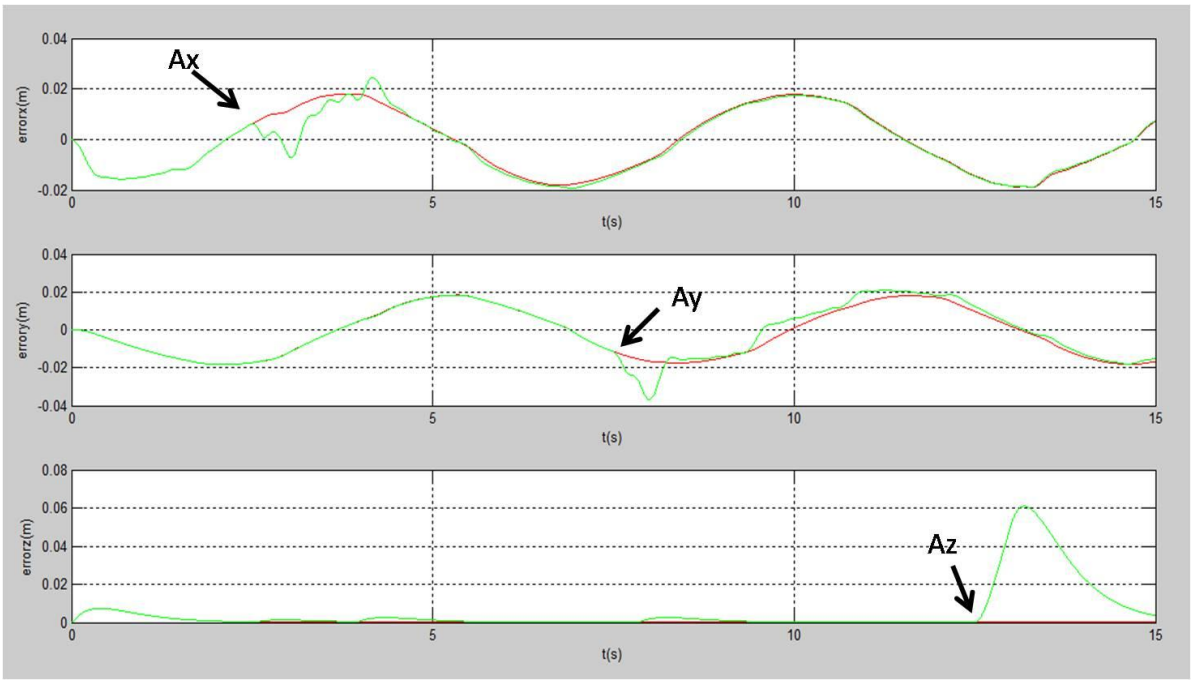



Figure 3.2-19 Trajectory tracking error with IB for an helicoidally trajectory

In the following two pictures, the desired angles that leave the IB controller are compared with the current angles in the cases of disturbances and no disturbances. In both cases, the system is able to follow the desired trajectory. Nevertheless, the introduction of the disturbance makes that the attitude trajectory that must be followed by the quadcopter ceases being sinusoidal to be sharper in order to recover the desired path.

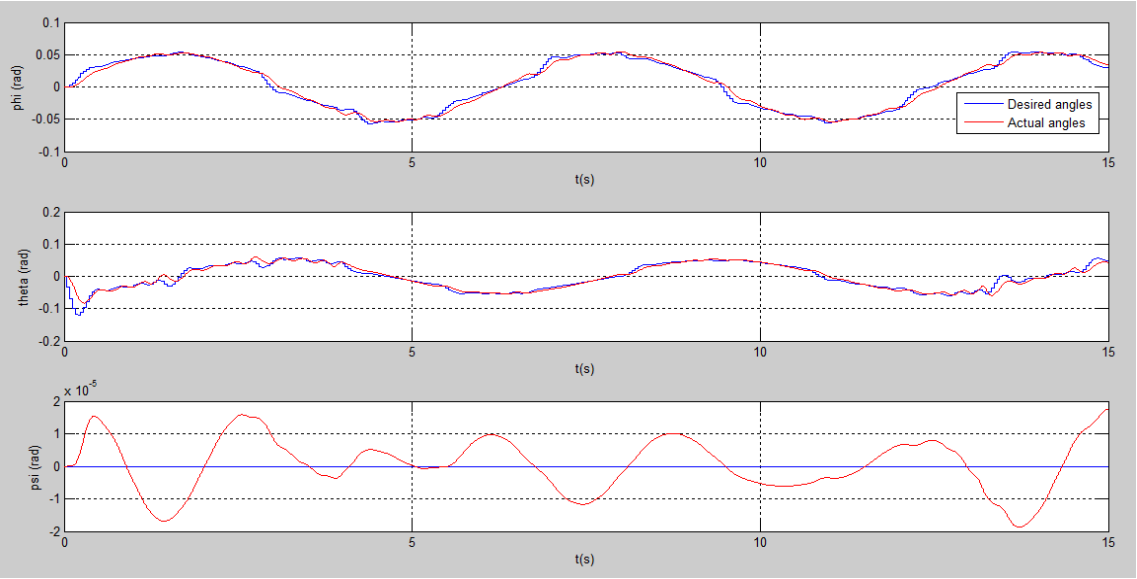



Figure 3.2-20 Desired and current attitude in IB controller without disturbances

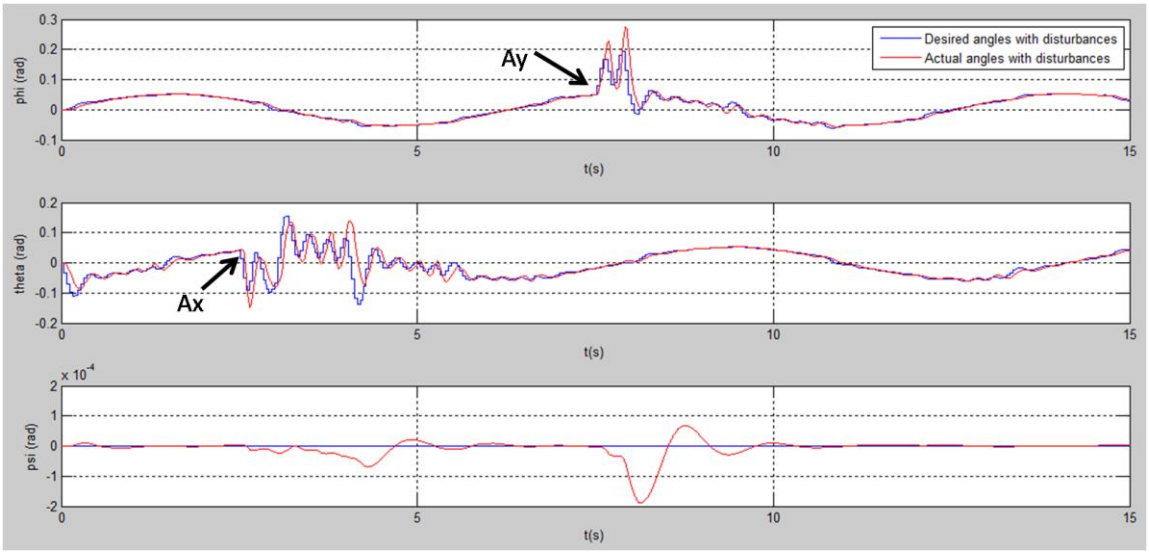



Figure 3.2-21 Desired and current attitude in IB controller with disturbances

To conclude, the values of $U_1$, $U_2$, $U_3$ and $U_4$ when there is no disturbance are shown. The objective is to compare them with the ones that will be displayed in fFigure 3.3-20.

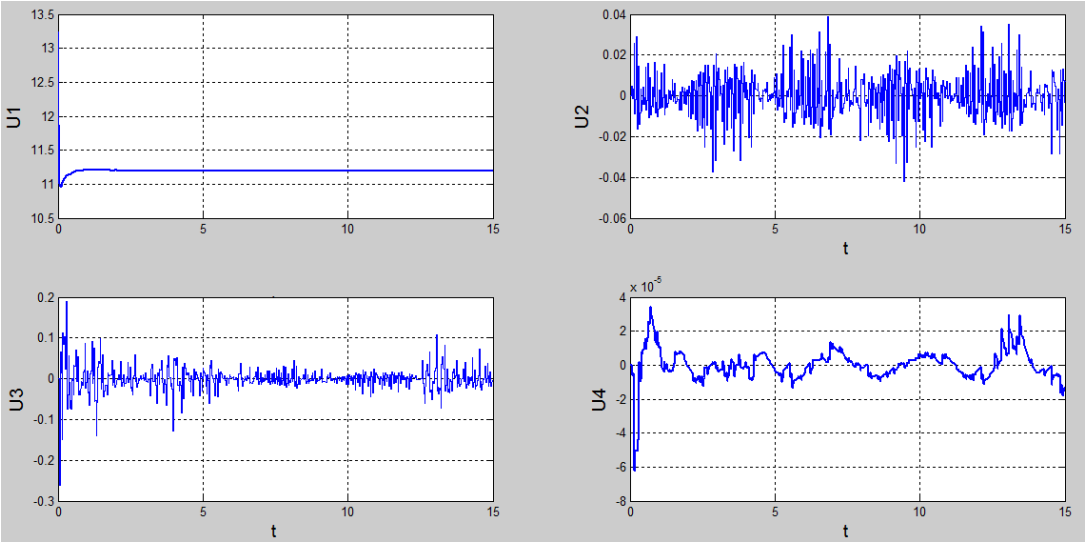



Figure 3.2-22 Total thrust and torques for an helicoidally entry with the IB controller

The constants used in this section could have been better adjusted by using optimization methods depending on whether the overshoot, the steady state error or the rise time must be smaller than a fixed value. Also, an area with many obstacles could require that the error to external disturbances is lower than obtained in order to maintain the integrity of the quadcopter. However, the current results achieved in this section are quite accurate and allow the control of the vehicle in the three axes as well as the yaw. When the control of the real vehicle is performed using this method, several tests would have to be performed to accurately adjust the variables before the first flight.

## 3.3 Control using an integral predictive/nonlinear $H_\infty$

In this section, another strategy to control the quadcopter is developed. The proposed structure consists of a model predictive controller which is able to obtain an adequate attitude and thrust given a desired path, and a nonlinear $H_\infty$ controller. The last one calculates the torques applied to the quadcopter; its inputs are the attitude angles and its derivations and double derivations. This method can follow a reference trajectory in a smooth way, through the use of the MPC and to hold up with disturbances by the used of the nonlinear $H_\infty$. The scheme of the model that will be described in detail in the following sections can be seen below.

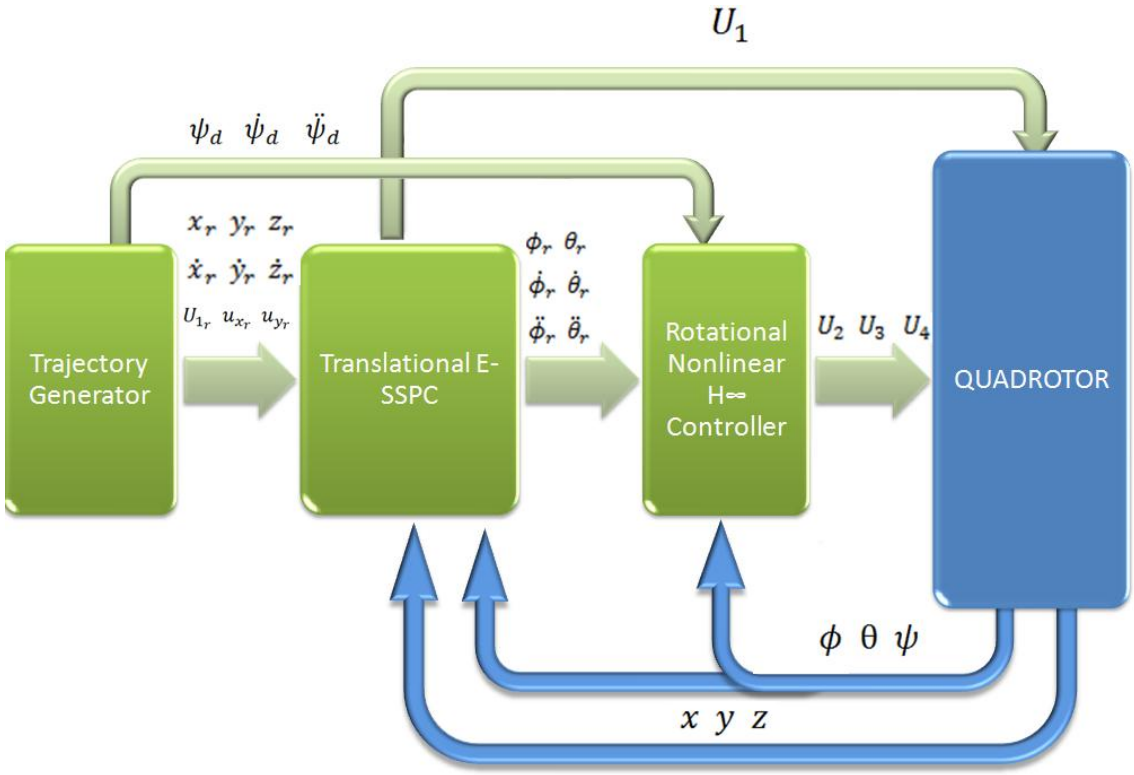



Figure 3.3-1 Diagram of the Integral predictive/nonlinear controller

For this controller, the model of the quadcopter is going to be slightly changed so it matches with the model used in the article (G. V. Raffo, M. G. Ortega, and F. R. Rubio., 2010c) in which is based this third controller. The altitude and position equations are the same as in (57) although the aerodynamic forces have been added as external disturbances.

$$\begin{aligned} \ddot{X} &= \frac{U_1}{m} \cdot (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) + \frac{A_x}{m} \\ \ddot{Y} &= \frac{U_1}{m} \cdot (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) + \frac{A_y}{m} \\ \ddot{Z} &= \frac{U_1}{m} \cdot (\cos\phi \, cos\theta) - g + \frac{A_z}{m} \end{aligned} \quad (106)$$

The changes in the mathematical model that describes the quadcopter rotational movement are more noticeable. The vector that contains the attitude angles will be defined as $\eta = [\phi \ \theta \ \psi]'$. The equation (10) is renamed as:

$$\omega = \boldsymbol{W_\eta} \cdot \dot{\eta} \quad (107)$$

The Lagrange method will be used in this case to solve the system. The rotational kinetic energy of the system is defined as:

$$E_c = \frac{1}{2}\dot{\eta}' J \dot{\eta} \quad (108)$$

where J is the Jacobian matrix: $J = W_\eta^{'} I W_\eta$ and I is the diagonal moment of inertia tensor. Introducing (108) in (34), the rotational equations can be written as:

$$M(\eta)\ddot{\eta} + C(\eta,\dot{\eta})\dot{\eta} = \tau \tag{109}$$

τ is composed by the addition of the control torques ($U_2$,$U_3$,$U_4$) and the external disturbances ($A_p$, $A_q$, $A_r$).

$$\tau = \tau_{\eta_a} + \tau_{\eta_d} \tag{110}$$

The matrices M and C are composed by the following terms:

$$M(\eta) = J(\eta) = \begin{bmatrix} I_x & 0 & -I_x s\theta \\ 0 & I_y c^2\phi + I_z s^2\phi & (I_y - I_z)c\phi s\phi c\theta \\ -I_x s\theta & (I_y - I_z)c\phi s\phi c\theta & I_x s^2\theta + I_y s^2\phi c^2\theta + I_z c^2\phi c^2\theta \end{bmatrix} \tag{111}$$

$$C(\eta,\dot{\eta}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

$$c_{11} = 0$$

$$c_{12} = (I_y - I_z)(\dot{\theta}c\phi s\phi + \dot{\psi}s^2\phi c\theta) + (I_z - I_y)\dot{\psi}c^2\phi c\theta - I_x\dot{\psi}c\theta = -c_{21}$$

$$c_{13} = (I_z - I_y)\dot{\psi}c\phi s\phi c^2\theta$$

$$c_{22} = (I_z - I_y)\dot{\phi}c\phi s\phi \tag{112}$$

$$c_{23} = -I_x\dot{\psi}s\theta c\theta + I_y\dot{\psi}s^2\phi s\theta c\theta + I_z\dot{\psi}c^2\phi s\theta c\theta$$

$$c_{31} = (I_y - I_z)\dot{\psi}c^2\theta s\phi c\phi - I_x\dot{\theta}c\theta$$

$$c_{32} = (I_z - I_y)(\dot{\theta}c\phi s\phi s\theta + \dot{\phi}s^2\phi c\theta) + (I_y - I_z)\dot{\phi}c^2\phi c\theta + I_x\dot{\psi}s\theta c\theta - I_y\dot{\psi}s^2\phi s\theta c\theta - I_z\dot{\psi}c^2\phi s\theta c\theta$$

$$c_{33} = (I_y - I_z)\dot{\psi}c\phi s\phi c^2\theta - I_y\dot{\theta}s^2\phi s\theta c\theta - I_z\dot{\theta}c^2\phi s\theta c\theta + I_x\dot{\theta}s\theta c\theta$$

### *3.3.1 Nonlinear controller*

There are many well-established analysis and design techniques for linear-time-invariant systems (LTI) such as root-locus, bode plot, Nyquist criterion, or pole placement. In control systems, however, there might be some elements that are not LTI thus these methods cannot be applied directly. The nonlinear theory studies how to approach these systems; although, the mathematical analyses needed to apply them successfully and to justify their conclusions are usually complex. In these methods, the control problem is seen as a mathematical optimization problem which looks for the controller that solves the optimization. Nevertheless, it must be bear in mind that the found controller is optimal with respect to the selected cost

function; but it might not be the best controller in regard to other measurements such as settling time, overshoot or energy expended. Sometimes, even if the LTI theory can be used to design a controller, a nonlinear one may be selected due to its simpler implementation, decreased of control energy or increased speed.

Beyond difficulties of controlling systems that have less control inputs than degrees of freedom, the chosen control law must be able to deal with model uncertainties. Because of this purpose, the nonlinear $H_\infty$ is introduced.

A nonlinear $H_\infty$ controller is used in the inner loop of the system and is in charge of the stabilization of the quadcopter. The system inputs the attitude angles as well as their derivatives and double derivatives in time, and outputs the torques applied in the three axes.

First of all, the tracking error vector needs to be defined. It is comprised of the error between the real attitude angles that come from the feedback and the desired ones. Moreover, the velocity error and the integral error are also included. The goal of adding this last term is achieving a null steady-state error as it was also done in the backstepping control.

$$\boldsymbol{x_\eta} = \begin{bmatrix} \dot{\tilde{\boldsymbol{\eta}}} \\ \tilde{\boldsymbol{\eta}} \\ \int \tilde{\boldsymbol{\eta}} dt \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\eta}} - \dot{\boldsymbol{\eta}}_{\boldsymbol{d}} \\ \boldsymbol{\eta} - \boldsymbol{\eta_d} \\ \int (\boldsymbol{\eta} - \boldsymbol{\eta_d})\, dt \end{bmatrix} \tag{113}$$

The control law used in (G. V. Raffo, M. G. Ortega, and F. R. Rubio., 2010c) for the rotational system is:

$$\boldsymbol{\tau}_{\eta_a} = \boldsymbol{M(\eta)\ddot{\eta}} + \boldsymbol{C(\eta,\dot{\eta})\dot{\eta}} - \underbrace{\boldsymbol{T_1}^{-1}}_{\rho \boldsymbol{I}} \left(\boldsymbol{M(\eta)T\dot{x}_\eta} + \boldsymbol{C(\eta,\dot{\eta})Tx_\eta}\right) + \boldsymbol{T_1}^{-1} u \tag{114}$$

where the matrix T is divided as $T = [T_1\ T_2\ T_3]$, $\rho$ is a positive constant and I is the identity matrix. The above equation can be simplified by using the vector of disturbances $\boldsymbol{d} = \boldsymbol{M(\eta)T\dot{x}_\eta M^{-1}(\eta)\tau}_{\eta_d}$ and introducing the equation in (109). With this procedure, the following formula is obtained:

$$\boldsymbol{M(\eta)T\dot{x}_\eta} + \boldsymbol{C(\eta,\dot{\eta})Tx_\eta} = \boldsymbol{u} + \boldsymbol{d} \tag{115}$$

A cost variable is defined as:

$$\xi = \boldsymbol{W} \begin{bmatrix} h(\boldsymbol{x}) \\ \boldsymbol{u} \end{bmatrix}$$

where W is a weighting matrix and u the vector control inputs. The optimization problem consists on finding the minimum value of γ for which exists a state **u** so that:

$$\int_0^T \|\boldsymbol{\xi}\|_2^2\, dt \leq \gamma \int_0^T \|\boldsymbol{d}\|_2^2\, dt$$

The left term can be expressed as:

$$\|\boldsymbol{\xi}\|_2^2 = \boldsymbol{\xi}'\boldsymbol{\xi} = [h'(\boldsymbol{x}) \quad \boldsymbol{u}']\boldsymbol{W}'\boldsymbol{W}\begin{bmatrix} h(\boldsymbol{x}) \\ \boldsymbol{u} \end{bmatrix} = [h'(\boldsymbol{x}) \quad \boldsymbol{u}']\begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} \\ \boldsymbol{S}' & \boldsymbol{R} \end{bmatrix}\begin{bmatrix} h(\boldsymbol{x}) \\ \boldsymbol{u} \end{bmatrix}$$

The generic dynamic equation of a nonlinear system can be expressed as follows:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, t) + g(\boldsymbol{x}, t)\boldsymbol{u} + k(\boldsymbol{x}, t)\boldsymbol{d} \tag{116}$$

The optimal control signal **u*** for this system can be obtain when the following HJBI equation has a smooth solution. The solution of a Hamilton-Jacobi-Bellman equation gives the minimum cost for a dynamical system with a determinate cost function.

$$\frac{\partial V}{\partial t} + \frac{\partial' V}{\partial \boldsymbol{x}} f(\boldsymbol{x}, t) + \frac{1}{2}\frac{\partial' V}{\partial \boldsymbol{x}}\left[\frac{1}{\gamma^2}k(\boldsymbol{x}, t)k'(\boldsymbol{x}, t) - g(\boldsymbol{x}, t)\boldsymbol{R}^{-1}g'(\boldsymbol{x}, t)\right]\frac{\partial V}{\partial \boldsymbol{x}} - \frac{\partial' V}{\partial \boldsymbol{x}}g(\boldsymbol{x}, t)\boldsymbol{R}^{-1}\boldsymbol{S}'h(\boldsymbol{x}) + \frac{1}{2}h'(\boldsymbol{x})\left(\boldsymbol{Q} - \boldsymbol{S}\boldsymbol{R}^{-1}\boldsymbol{S}'\right)h(\boldsymbol{x}) = 0$$

In this case, the optimal control signal is given by:

$$u^* = -\boldsymbol{R}^{-1}\left(\boldsymbol{S}'h(\boldsymbol{x}) + g'(\boldsymbol{x}, t)\frac{\partial V(\boldsymbol{x}, t)}{\partial \boldsymbol{x}}\right) \tag{117}$$

To use this result in the quadcopter model, the equation (115) needs to be rewritten in the terms of (116). After defining $h(\boldsymbol{x})$ as the error vector and some manipulations, the optimal control law is written as:

$$\boldsymbol{\tau}^*{}_{\eta_a} = \boldsymbol{M}(\boldsymbol{\eta})\ddot{\boldsymbol{\eta}}_{\boldsymbol{r}} + \boldsymbol{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}} - \boldsymbol{M}(\boldsymbol{\eta})(\boldsymbol{K_D}\dot{\tilde{\boldsymbol{\eta}}} + \boldsymbol{K_P}\tilde{\boldsymbol{\eta}} - \boldsymbol{K_I}\int \tilde{\boldsymbol{\eta}}dt) \tag{118}$$

The constants matrices are given in the equation below where the terms $\omega_1, \omega_2, \omega_3$ and $\omega_u$ are tuned in a similar way as it was done with the PID controller.

$$\begin{aligned} \boldsymbol{K_D} &= \frac{\sqrt{{\omega_2}^2 + 2\omega_1\omega_3}}{\omega_1}\boldsymbol{I} + \boldsymbol{M}(\boldsymbol{\eta})^{-1}\left(\boldsymbol{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) + \frac{1}{{\omega_u}^2}\boldsymbol{I}\right) \\ \boldsymbol{K_P} &= \frac{\omega_3}{\omega_1}\boldsymbol{I} + \frac{\sqrt{{\omega_2}^2 + 2\omega_1\omega_3}}{\omega_1}\boldsymbol{M}(\boldsymbol{\eta})^{-1}\left(\boldsymbol{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) + \frac{1}{{\omega_u}^2}\boldsymbol{I}\right) \\ \boldsymbol{K_D} &= \frac{\omega_3}{\omega_1}\boldsymbol{M}(\boldsymbol{\eta})^{-1}\left(\boldsymbol{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) + \frac{1}{{\omega_u}^2}\boldsymbol{I}\right) \end{aligned} \tag{119}$$

The control law design of these controllers can only control the same quantity of degrees of freedom as inputs it has. The remaining DOF are assumed to have a zero stable dynamic or they are controlled in an outer-loop. For this reason, the position and altitude control will be achieved by using a model predictive controller as it will be seen in the next section.

### 3.3.2 Model Predictive Control

Model Predictive Control (MPC) is an advanced, optimization based control strategy applicable to a wide range of industrial applications such as chemical plants and internal combustion engines. In this controller, a model of the system is used to predict the future evolution of the process that along with the current measurements can optimize the control signal. Having the ability to anticipate future events, it can accordingly take control actions.

The resulting optimization problem from the control design needs to be solved at each sampling interval, based on actual measurements. In the past, the computationally demanding nature of algorithms, solving these types of control problems, prevented the application of MPC for embedded systems. The computational power of the CPUs of the embedded systems has increased dramatically. Fast and reliable solvers can be used in embedded systems; therefore, this problem has been removed.

The main objectives of the MPC are:

- Prevent violations of input and output constrains.
- Drive some output variables to their optimal set points, while maintaining other outputs within specified ranges.
- Prevent excessive movement of the input variables.

The procedure is as follows: at the k-th sampling instant, the values of the manipulated variables, u, at the next M (control horizon) sampling instants, {u (k), u (k+1), …, u (k+M -1)} are calculated. This set of M values is calculated so as to minimize the predicted deviations from the reference trajectory over the next P (prediction horizon) sampling instants while satisfying the constraints. Then, u (k) is implemented. At the next sampling instant, k+1, u is re-calculated for the next M sampling instants, from k+1 to k+M, and u(k+1) is implemented. This procedure is repeated for subsequent sampling instants.

Two state-space MPC based on the error model have been performed. The first one controls the altitude, while the second one controls the x and y motion. The quadcopter model (106) is rewritten in a state-space form:

$$\dot{\boldsymbol{\xi}}(t) = f\left(\boldsymbol{\xi}(\boldsymbol{t}), \boldsymbol{u}_{\boldsymbol{\xi}}(t)\right) = \begin{bmatrix} u_0(t) \\ u_x(t)\dfrac{U_1(t)}{m} + \dfrac{A_x(t)}{m} \\ v_0(t) \\ u_y(t)\dfrac{U_1(t)}{m} + \dfrac{A_y(t)}{m} \\ w_0(t) \\ -g + (cos\theta cos\phi)\dfrac{U_1(t)}{m} + \dfrac{A_z(t)}{m} \end{bmatrix} \quad (120)$$

$$\begin{aligned} \bar{u}_x &= \cos\phi_r \sin\theta_r \cos\psi + \sin\phi_r \sin\psi \\ \bar{u}_y &= \cos\phi_r \sin\theta_r \sin\psi - \sin\phi_r \cos\psi \end{aligned} \quad (121)$$

where $\bar{u}_x$ and $\bar{u}_y$ can be considered as the direction of the total thrust $U_1$. To obtained the future predictive values, a virtual reference vehicle with the same behavior as the actual quadcopter will be used. This vehicle is supposed to be stabilized in altitude and to have null external disturbances. As the desired position and altitude are the inputs of the whole system, the following control inputs of the reference vehicle can be calculated:

$$U_{1_r} = m \cdot (\ddot{z}_r + g) \qquad u_{x_r} = \frac{\ddot{x}_r \cdot m}{U_{1_r}} \qquad u_{y_r} = \frac{\ddot{y}_r \cdot m}{U_{1_r}} \tag{122}$$

The error model which will be denoted hereinafter with the symbol ~, can be defined by subtracting the virtual reference vehicle from the actual system. Moreover, the integral of the position term has also be added to state-space vector of the system, as it has also been done before in this report, to achieve null steady-state error.

$$x_\varepsilon(t) = \begin{bmatrix} \tilde{x}(t) \\ \tilde{u}_0(t) \\ \int \tilde{x}(t)dt \\ \tilde{y}(t) \\ \tilde{v}_0(t) \\ \int \tilde{y}(t)dt \\ \tilde{z}(t) \\ \tilde{v}_0(t) \\ \int \tilde{z}(t)dt \end{bmatrix} = \begin{bmatrix} x(t) - x_r(t) \\ u_0(t) - u_{0_r}(t) \\ \int (x(t) - x_r(t))\,dt \\ y(t) - y_r(t) \\ v_0(t) - v_{0_r}(t) \\ \int (y(t) - y_r(t))\,dt \\ z(t) - z_r(t) \\ w_0(t) - w_{0_r}(t) \\ \int (z(t) - z_r(t))\,dt \end{bmatrix} \tag{123}$$

If the model is discretized, being A and B the Jacobians of the system (120), it is obtained:

$$\boldsymbol{x_\xi}(k+1) = \boldsymbol{A} \cdot \boldsymbol{x_\xi}(k) + \boldsymbol{B} \cdot \tilde{\boldsymbol{u}}_\xi(k) \tag{124}$$

As it has been said, it will be two controllers, one for the altitude and the other for the position. With this purpose, the above model is divided in two.

$$A_z = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ \Delta t & 0 & 1 \end{bmatrix} \qquad B_z = \begin{bmatrix} 0 \\ \frac{\Delta t}{m} cos\theta(k) cos\phi(k) \\ 0 \end{bmatrix} \tag{125}$$

$$A_{xy} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \Delta t & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \Delta t & 0 & 1 \end{bmatrix} \qquad B_{xy} = \begin{bmatrix} 0 & 0 \\ \frac{\Delta t}{m} U_1(k) & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{\Delta t}{m} U_1(k) \\ 0 & 0 \end{bmatrix} \tag{126}$$

The predictions of the model are denoted by $\hat{x}_\varepsilon(k+j|k)$, which is a prediction for time *k+j* being at time *k*. A state-space model at time k that takes into account the predictions is shown below.

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\cdot\boldsymbol{x}(k) + \boldsymbol{B}(k)\cdot\boldsymbol{u}(k)$$

$$\boldsymbol{x}(k+2) = \boldsymbol{A}\cdot\boldsymbol{x}(k+1) + \boldsymbol{B}(k+1)\cdot\boldsymbol{u}(k+1)$$

Introducing $\boldsymbol{x}(k+1)$ in the last equation, it is obtained:

$$\boldsymbol{x}(k+2) = \boldsymbol{A}^2\cdot\boldsymbol{x}(k) + \boldsymbol{AB}(k)\cdot\boldsymbol{u}(k) + \boldsymbol{B}(k+1)\cdot\boldsymbol{u}(k+1)$$

The procedure is repeated N times:

$$\boldsymbol{x}(k+N) = \boldsymbol{A}^N\cdot\boldsymbol{x}(k) + \boldsymbol{A}^{\boldsymbol{N-1}}\boldsymbol{B}(k)\cdot\boldsymbol{u}(k) + \boldsymbol{A}^{\boldsymbol{N-2}}\boldsymbol{B}(k+1)\cdot\boldsymbol{u}(k+1) + \cdots +$$

$$+\boldsymbol{AB}(k+N-2)\cdot\boldsymbol{u}(k+N-2) + \boldsymbol{B}(k+N-1)\cdot\boldsymbol{u}(k+N-1)$$

The complete system at time k is:

$$\begin{bmatrix}\hat{x}_{k+1}\\ \hat{x}_{k+2}\\ \vdots\\ \hat{x}_{k+N}\end{bmatrix} = \underbrace{\begin{bmatrix}A\\ A^2\\ \vdots\\ A^N\end{bmatrix}}_{P}\cdot x_k + \underbrace{\begin{bmatrix}B_k & 0 & \cdots & 0\\ B_k & B_{k+1} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ B_k & B_{k+1} & \cdots & B_{k+N-1}\end{bmatrix}}_{H}\cdot\begin{bmatrix}\hat{\tilde{u}}_1\\ \hat{\tilde{u}}_2\\ \vdots\\ \hat{\tilde{u}}_{k+N-1}\end{bmatrix} \quad (127)$$

The altitude controller will be developed below. The procedure for the other controller is quite similar. The system (127) for the altitude controller can be expressed as:

$$\hat{\boldsymbol{x}}_{\boldsymbol{\xi z}} = \boldsymbol{P}_z(k|k)\cdot\boldsymbol{x}_{\boldsymbol{\xi z}}(k|k) + \boldsymbol{H}_z(k|k)\cdot\hat{\tilde{\boldsymbol{u}}}_{\boldsymbol{\xi z}} \quad (128)$$

The purpose of the controller is to find the control inputs that can make the error between the reference values and the actual values obtained by the sensor equal to zero: $\lim_{t\to\infty}\boldsymbol{x}_{\boldsymbol{\xi}} = 0$. This is achieved by minimizing the cost in equation (128) where Q and R are weighting diagonal definite positive matrices whose values will have to be determinate; and N is the prediction horizon.

$$J_z = \left[\hat{x}_{\xi z} - \hat{x}_{\xi r z}\right]' Q_z \left[\hat{x}_{\xi z} - \hat{x}_{\xi r z}\right] + \left[\hat{\tilde{u}}_{\xi z} - \hat{\tilde{u}}_{\xi r z}\right]' R_z \left[\hat{\tilde{u}}_{\xi z} - \hat{\tilde{u}}_{\xi r z}\right] + \Omega\left(\hat{x}_{\xi z}(k+N_z|k) - \hat{x}_{\xi r z}(k+N_z|k)\right) \quad (129)$$

After minimizing the equation, the system control law can be obtained:

$$\hat{\tilde{\boldsymbol{u}}}_{\varepsilon z} = [\boldsymbol{H'}_z\boldsymbol{Q}_z\boldsymbol{H}_z + \boldsymbol{R}_z]^{-1}\cdot[\boldsymbol{H'}_z\boldsymbol{Q}_z(\hat{\boldsymbol{x}}_{\varepsilon r z} - \boldsymbol{P}_z\boldsymbol{x}_{\varepsilon z}(k))] + \boldsymbol{R}_z\hat{\tilde{\boldsymbol{u}}}_{\varepsilon r z} \quad (130)$$

$$\hat{\boldsymbol{x}}_{\varepsilon r z} = \begin{bmatrix}\boldsymbol{x}_{\varepsilon r z}(k+1|k) - \boldsymbol{x}_{\varepsilon r z}(k|k)\\ \vdots\\ \boldsymbol{x}_{\varepsilon r z}(k+N|k) - \boldsymbol{x}_{\varepsilon r z}(k|k)\end{bmatrix} \quad (131)$$

$$\hat{\tilde{\boldsymbol{u}}}_{\varepsilon r z} = \begin{bmatrix}U_{1_r}(k|k) - U_{1_r}(k-1|k)\\ \vdots\\ U_{1_r}(k+N_u-1|k) - U_{1_r}(k-1|k)\end{bmatrix} \quad (132)$$

The value $\hat{\tilde{\boldsymbol{u}}}_{\varepsilon z}$ at time k, which is obtained from equation (130), is added to the total thrust $U_{1_r}$ given by (122), obtaining the control signal that must be applied to the real quadcopter.

$$U_1(k) = \widehat{\boldsymbol{u}}_{\varepsilon z}(k|k) + U_{1_r}(k) \tag{133}$$

The same method is carried out for the second controller, obtaining the same control law:

$$\widehat{\widehat{\boldsymbol{u}}}_{\varepsilon xy} = \left[\boldsymbol{H'}_{xy}\boldsymbol{Q}_{xy}\boldsymbol{H}_{xy} + \boldsymbol{R}_{xy}\right]^{-1} \cdot \left[\boldsymbol{H'}_{xy}\boldsymbol{Q}_{xy}\left(\hat{\boldsymbol{x}}_{\varepsilon rxy} - \boldsymbol{P}_{xy}\boldsymbol{x}_{\varepsilon xy}(k)\right)\right] + \boldsymbol{R}_{xy}\widehat{\widehat{\boldsymbol{u}}}_{\varepsilon rxy} \tag{134}$$

In this case, $\widehat{\widehat{\boldsymbol{u}}}_{\varepsilon xy}(k|k) = \left[\tilde{u}_x(k|k)\ \ \tilde{u}_y(k|k)\right]'$. This value is added to the reference values $u_{x_r}$ and $u_{y_r}$ calculated in (130) obtaining the following control signals:

$$\begin{bmatrix}\bar{u}_x(k)\\ \bar{u}_y(k)\end{bmatrix} = \begin{bmatrix}\bar{u}_x(k|k)\\ \bar{u}_y(k|k)\end{bmatrix} + \begin{bmatrix}u_{x_r}(k)\\ u_{y_r}(k)\end{bmatrix} \tag{135}$$

However, the desired outputs of this controller are the roll and pitch angles. These values are obtained by clearing the equations (121).

$$\begin{aligned}\phi_d &= asin\left[u_x \cdot sin\psi - u_y \cdot cos\psi\right]\\ \theta_d &= asin\left[\frac{u_y + cos\psi \cdot sin\left[asin\left[u_x \cdot sin\psi - u_y \cdot cos\psi\right]\right]}{sin\psi \cdot cos\left[asin\left[u_x \cdot sin\psi - u_y \cdot cos\psi\right]\right]}\right]\end{aligned} \tag{136}$$

### *3.3.3 Implementation in Simulink*

In this section, the two controllers developed in the previous section are designed in Simulink. The procedure to simulate this system is composed of three parts. The first one is a Simulink model of the trajectory generator. In the second part, the outputs of the trajectory generator are reallocated using a Matlab program, to be properly introduced in the third part. This last part is formed by the controllers and the quadcopter model.

- TRAJECTORY GENERATION

The inputs of this block are the desired position and psi angle. Its objective is to create a virtual reference vehicle, so predictive values can be later introduced in the controller.

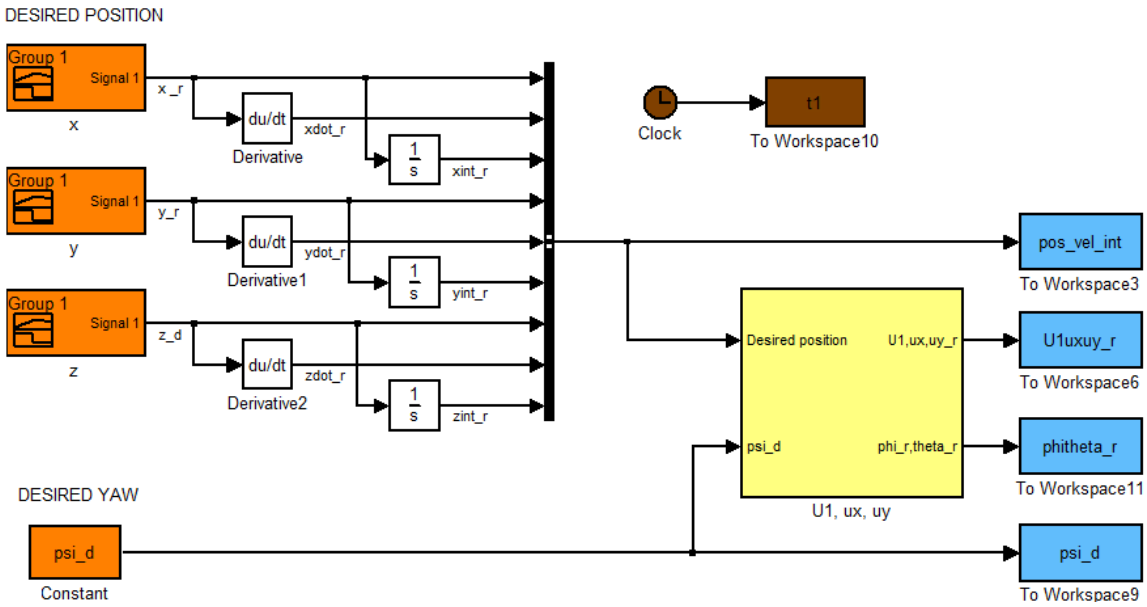



Figure 3.3-2 Trajectory generation block for the integral predictive/nonlinear controller

The positions are derived and integrated; then they are sent to the workspace. In this block, the save format "Structure with time" must be selected. This way, the 9th order vector is stored together with its time. All the values must be saved every $\Delta t$. This is accomplished by selecting a fixed step solver in the Configuration Parameters menu.

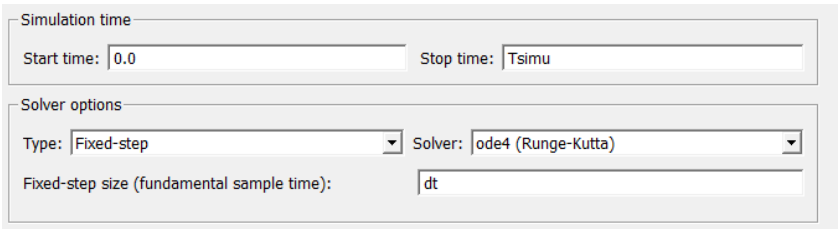



Figure 3.3-3 Configuration Parameters menu

In the yellow block, some reference values for the virtual vehicle that will be needed later are calculated. This way, the position, attitude, U1, $u_x$ and $u_y$ of the reference vehicle are known.

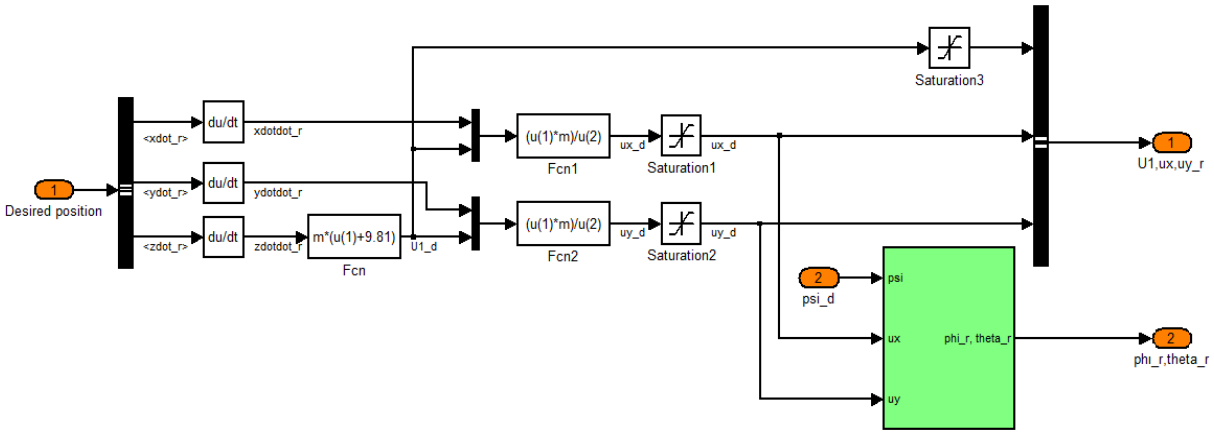



Figure 3.3-4 $U_1$, $u_x$ and $u_y$ calculations

The boxed functions are the ones in (122). As the future reference values of phi and theta are also included in the mathematic model of the MPC controller, these angles are calculated using the equations (136).

- REALLOCATION OF THE VARIABLES

Once all the equations needed in the MPC have been introduced in section 3.3.2, it must be studied how to introduce the values presented in the controller that come from the trajectory generator, in an appropriate manner.

The prediction and control horizons present in (131) and (132) have been chosen equal.

$$N_z = N_{u_z} = 3 \qquad\qquad N_{xy} = N_{u_{xy}} = 10$$

The procedure will be explained for the equation (131), for the remaining cases the procedure is similar. As it can be observed, this matrix at time k needs the actual value and the three following ones (k+1, k+2, k+3). The position values as well as their derivations and integrations have been saved in the workspace as follows:

| | $x$ | $\frac{dx}{dt}$ | $\int xdt$ | $y$ | $\frac{dy}{dt}$ | $\int ydt$ | z | $\frac{dz}{dt}$ | $\int zdt$ |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 5.0000e-04 | 0.5000 | 2.5000e-07 | 5.0000e-04 | 0.5000 | 2.5000e-07 | 5.0000e-04 | 0.5000 | 2.5000e-07 |
| 3 | 1.0000e-03 | 0.5000 | 1.0000e-06 | 1.0000e-03 | 0.5000 | 1.0000e-06 | 1.0000e-03 | 0.5000 | 1.0000e-06 |
| 4 | 0.0015 | 0.5000 | 2.2500e-06 | 0.0015 | 0.5000 | 2.2500e-06 | 0.0015 | 0.5000 | 2.2500e-06 |
| 5 | 0.0020 | 0.5000 | 4.0000e-06 | 0.0020 | 0.5000 | 4.0000e-06 | 0.0020 | 0.5000 | 4.0000e-06 |
| 6 | 0.0025 | 0.5000 | 6.2500e-06 | 0.0025 | 0.5000 | 6.2500e-06 | 0.0025 | 0.5000 | 6.2500e-06 |
| 7 | 0.0030 | 0.5000 | 9.0000e-06 | 0.0030 | 0.5000 | 9.0000e-06 | 0.0030 | 0.5000 | 9.0000e-06 |
| 8 | 0.0035 | 0.5000 | 1.2250e-05 | 0.0035 | 0.5000 | 1.2250e-05 | 0.0035 | 0.5000 | 1.2250e-05 |
| 9 | 0.0040 | 0.5000 | 1.6000e-05 | 0.0040 | 0.5000 | 1.6000e-05 | 0.0040 | 0.5000 | 1.6000e-05 |
| 10 | 0.0045 | 0.5000 | 2.0250e-05 | 0.0045 | 0.5000 | 2.0250e-05 | 0.0045 | 0.5000 | 2.0250e-05 |

k

Figure 3.3-5 Grouping of future values

In this case, only the columns 7 to 9 need to be considered. A new structure with time will be created called *z_rows.* Each time instant corresponds to a 1x12 matrix. It is formed by the elements of the previous green square left to right and top to bottom, that will be then introduced in the other Simulink model. The matlab program can be seen in Figure 3.3-6. The first step is to determinate the time of the new structure, that will not change. After that, a matrix called D with 4x3 dimensions is defined for each time. The matrix is reshape and then introduced in *z_row*.

```
z_rows.time=pos_vel_int.time;

for i=1:1:((Tsimu/dt)-2);
D = [];
D(4,:)=pos_vel_int.signals.values(i+3,7:9);
D(3,:)=pos_vel_int.signals.values(i+2,7:9);
D(2,:)=pos_vel_int.signals.values(i+1,7:9);
D(1,:)=pos_vel_int.signals.values(i,7:9);
D=D';
D=reshape(D,[1 12]);

z_rows.signals.values(i,:)= D;
end

D = [];
D(4,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(3,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(2,:)=pos_vel_int.signals.values((Tsimu/dt),7:9);
D(1,:)=pos_vel_int.signals.values((Tsimu/dt)-1,7:9);
D=D';
D=reshape(D,[1 12]);
z_rows.signals.values((Tsimu/dt)-1,:)= D;

D = [];
D(4,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(3,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(2,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(1,:)=pos_vel_int.signals.values((Tsimu/dt),7:9);
D=D';
D=reshape(D,[1 12]);
z_rows.signals.values((Tsimu/dt),:)= D;

D = [];
D(4,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(3,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(2,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D(1,:)=pos_vel_int.signals.values((Tsimu/dt)+1,7:9);
D=D';
D=reshape(D,[1 12]);
z_rows.signals.values((Tsimu/dt)+1,:)= D;
```

Figure 3.3-6 Matlab Code

The procedure is the same for the rest of the variables. It only has to be taken into account that for the variable $U_{1_r}$, the instant k-1 must also be included and as the horizon is bigger in xy controller, more values will have to be stored.

- CONTROLLERS AND QUADCOPTER MODEL

After executing the previous matlab file, this second model can be simulated. It is composed of three big subsystems, the quadcopter, the rotational non-linear $H_\infty$ controller and the E-SSPC controller. This model is shown in Figure 3.3-7 and it is run using a variable step solver. This means that when the signal changes a lot in a small period of time more values will be recorded than if the output is a constant. The reference values coming from the workspace correspond to equally spaced time instants. However, this is not a problem because the missing needed values are interpolated from the reference.

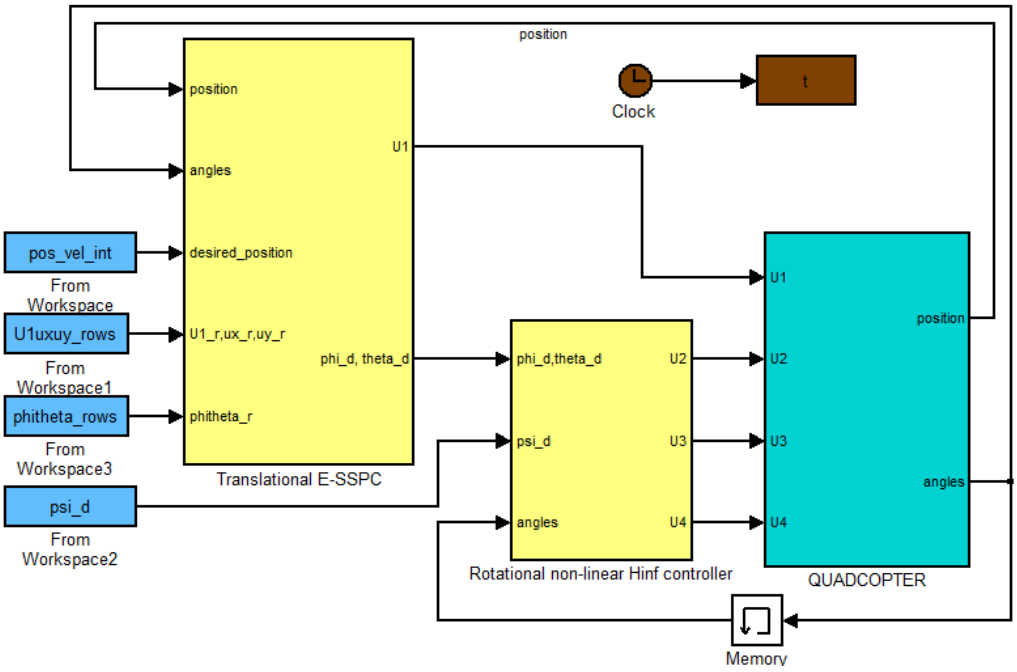



Figure 3.3-7 Integral predictive/nonlinear controller in Simulink

The attitude model of the quadcopter has been modified. The new attitude equations where presented in equations (111) and (112).

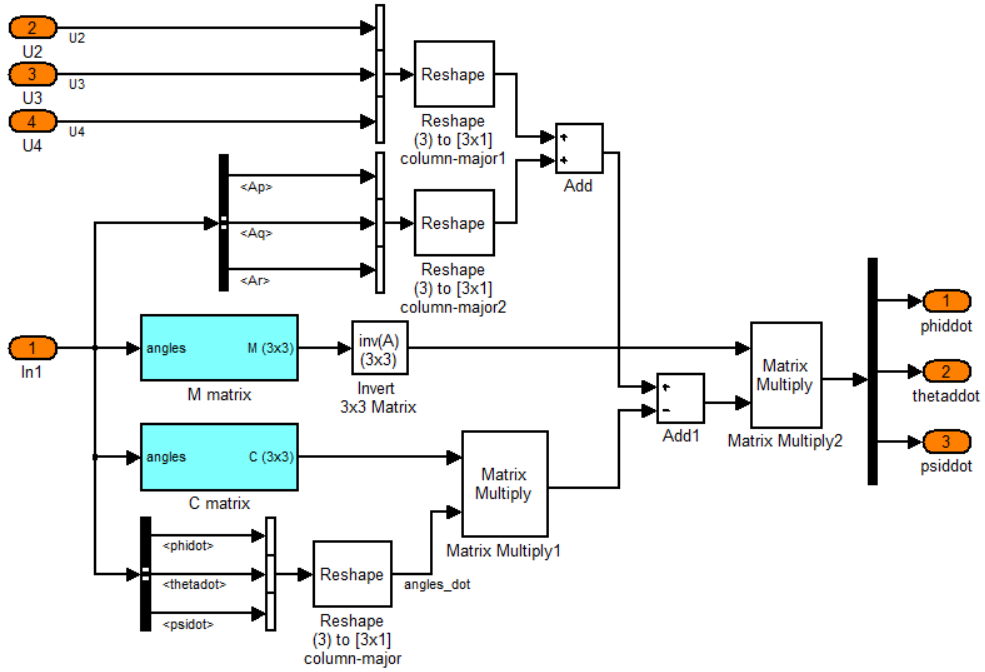



Figure 3.3-8 Attitude model of the quadcopter

To introduce the matrices M and C the block “Create a 3x3 matrix” has been used. It doesn’t exist a specific block to create matrices of other dimensions. To create them, it can be used the

block “vector concatenate” followed by a “Reshape” block that can be modified to the required dimensions. It is only needed to note that the values introduced in the “vector concatenate” which then will form the matrix, have to be entered by rows.

The subsystem of the rotational non-linear H∞ controller is detailed below. The system (118) must be represented in the block. The entrances are the current angles and its derivations and the desired angles from the MPC controller. The subtraction of this to vectors generates the error vector η. The gain matrices are calculated in the green blocks using equation (119).

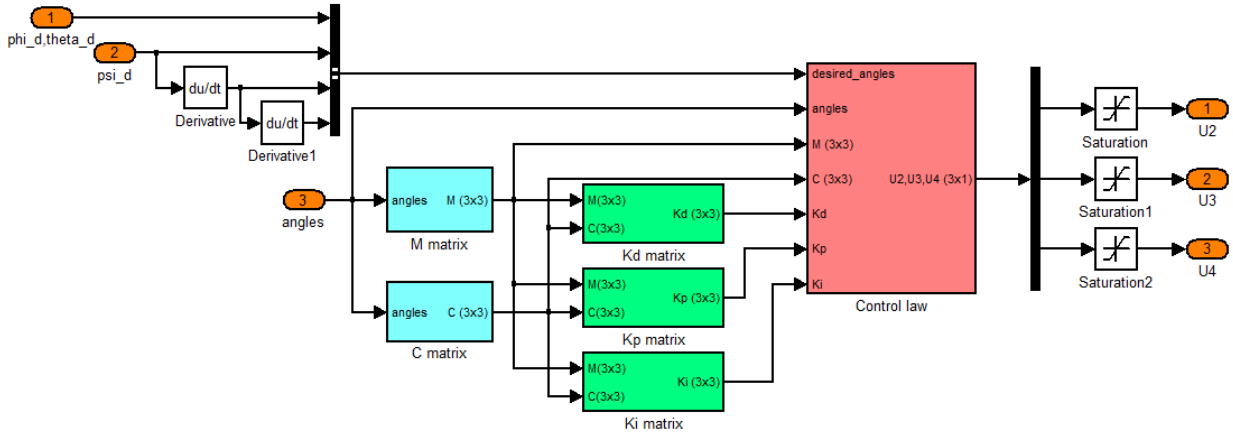



Figure 3.3-9 Rotational non-linear H∞ controller in Simulink

The block containing the control law is represented below:

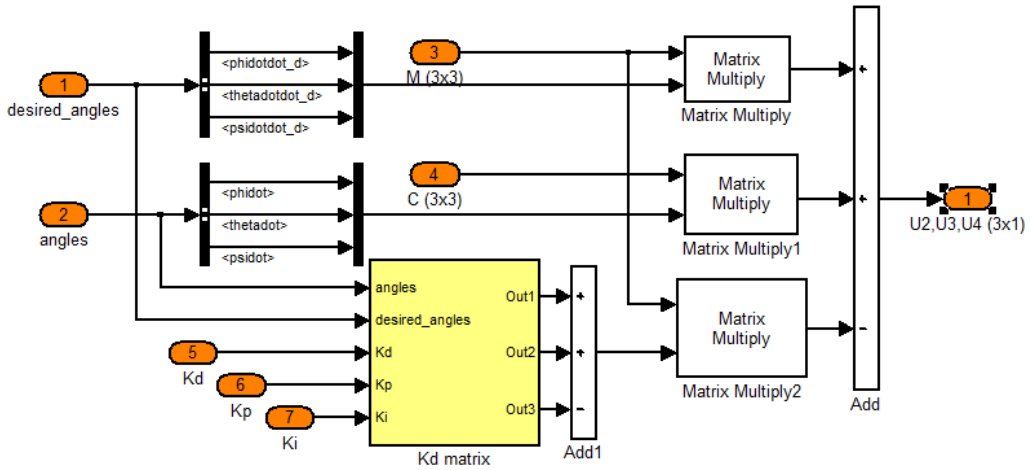



Figure 3.3-10 Control law of the rotational non-linear H∞ controller

The last block, the Translational E-SSPC controller is detailed below.

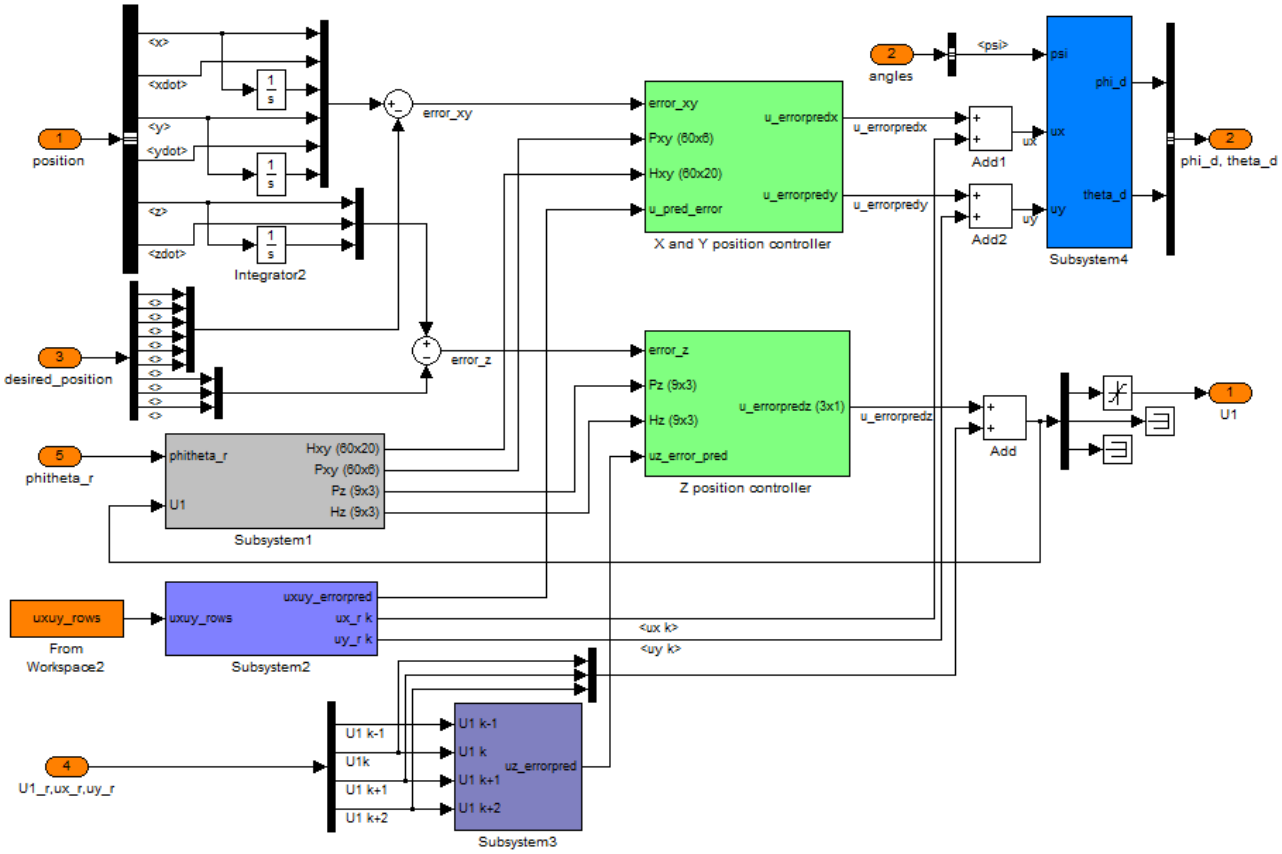



Figure 3.3-11 E-SSPC controller subsytem in Simulink

In the grey block, the matrices H and P are calculated by using the matrices A and B.

The purple blocks are the ones in charge of reproducing the equation (132) and the corresponding to xy. As for the green blocks, they include the equations (130) and (134). Figure 3.3-12 reproduces the Z position controller block.

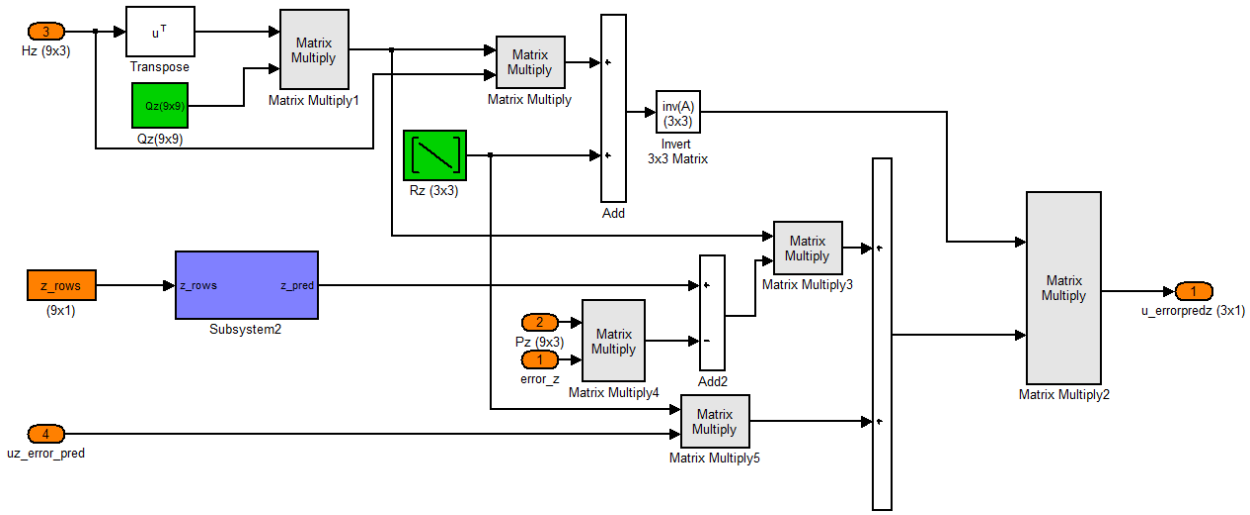



Figure 3.3-12 Z position controller subsystem

In the last figure, the purple block, converts the data from the matlab file that has been explained before into the vector (131), as it can be seen in the following figure.

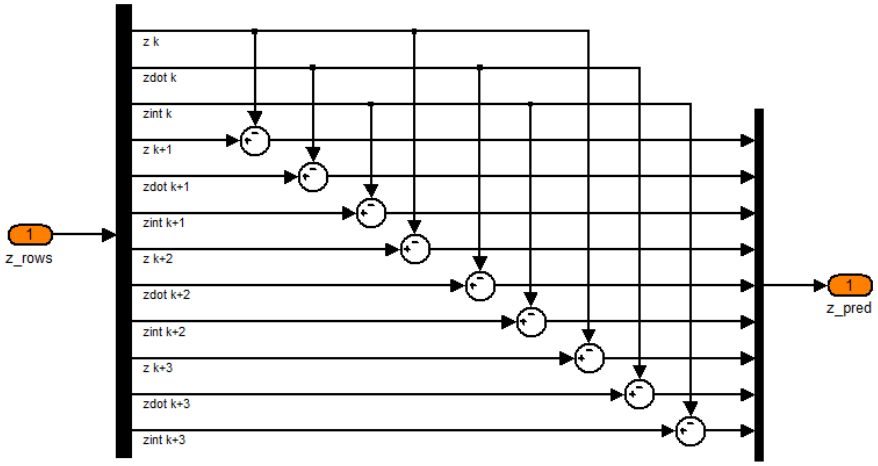



Figure 3.3-13 Data conversion

### 3.3.4 *Simulation results*

Following the instructions on the previous section, the integral predictive and nonlinear controller is simulated. The controller has several constants that must be set in order to obtain an adequate response. This task might be hard if it is not realize with some order. The constants are related to each other and in some of them a small change in can cause the system to move from a good outcome to a completely wrong one. The best way is to adjust the constants involved in the inner loop first. Only the nonlinear controller is simulated when the total thrust is a constant and the entries are the three desired angles. In the following picture, the simulated system is represented:

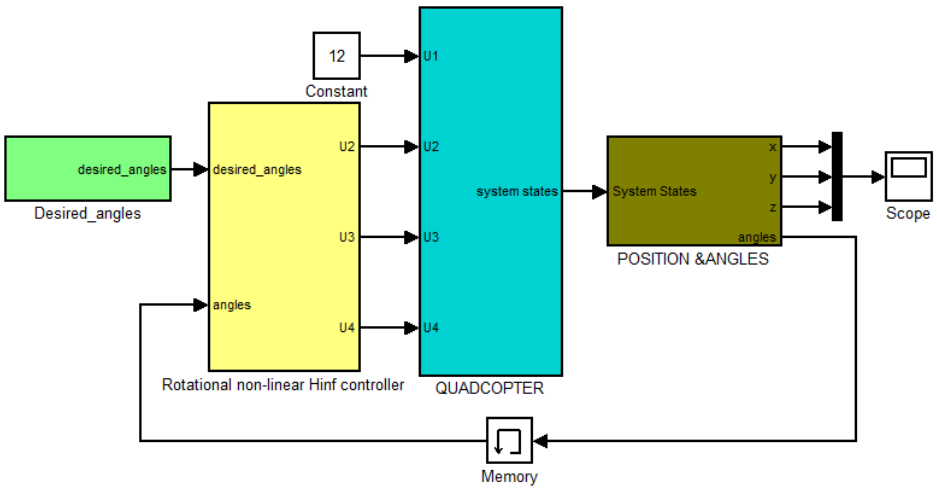



Figure 3.3-14 Nonlinear controller only

Using this system, the variables $w_1$, $w_2$, $w_3$ and $w_u$ are tuned. The observation of equation (119), can help to tune the values: the increase in $w_2$ and $w_3$ generates an augmentation in all

the constants (proportional, derivative and integral) and an increase in $w_1$ and $w_u$ a fall. The selected values have been:

| $w_1$ | $w_2$ | $w_3$ | $w_u$ |
|---|---|---|---|
| 5000 | 0,5 | 100 | 500 |

Table 7: Nonlinear controller selected constants

The above values are introduced in the system. In Figure 3.3-15, the entries of the system are reproduced as well as the outputs. In the next picture, the magnitude of the error can be observed. As it can be seeing, the entry is accurately reproduced.

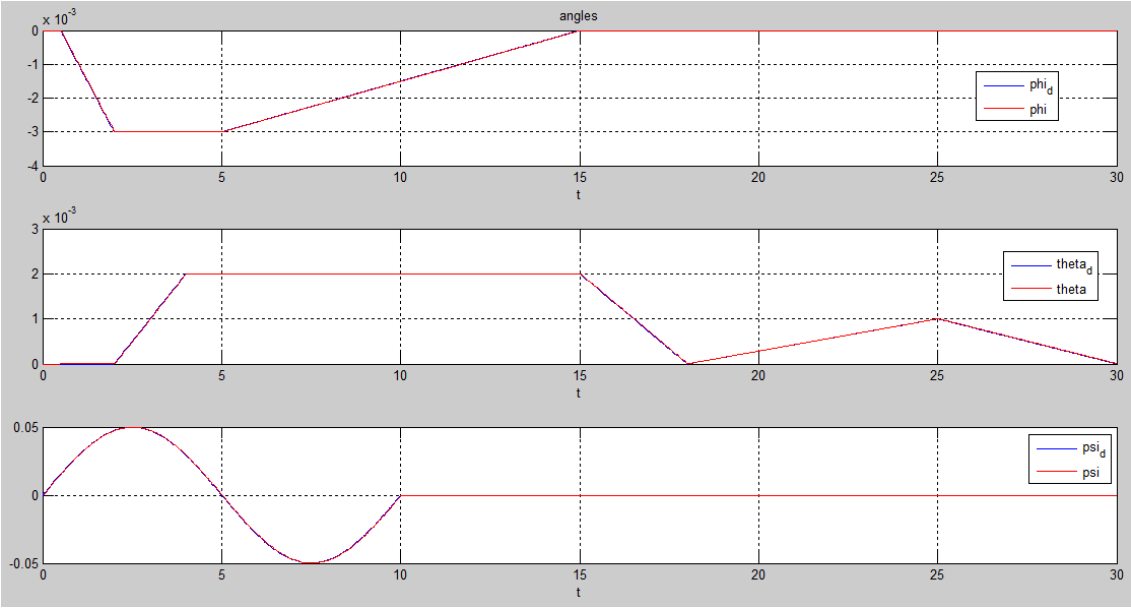



Figure 3.3-15 Desired and current attitude for the nonlinear controller only

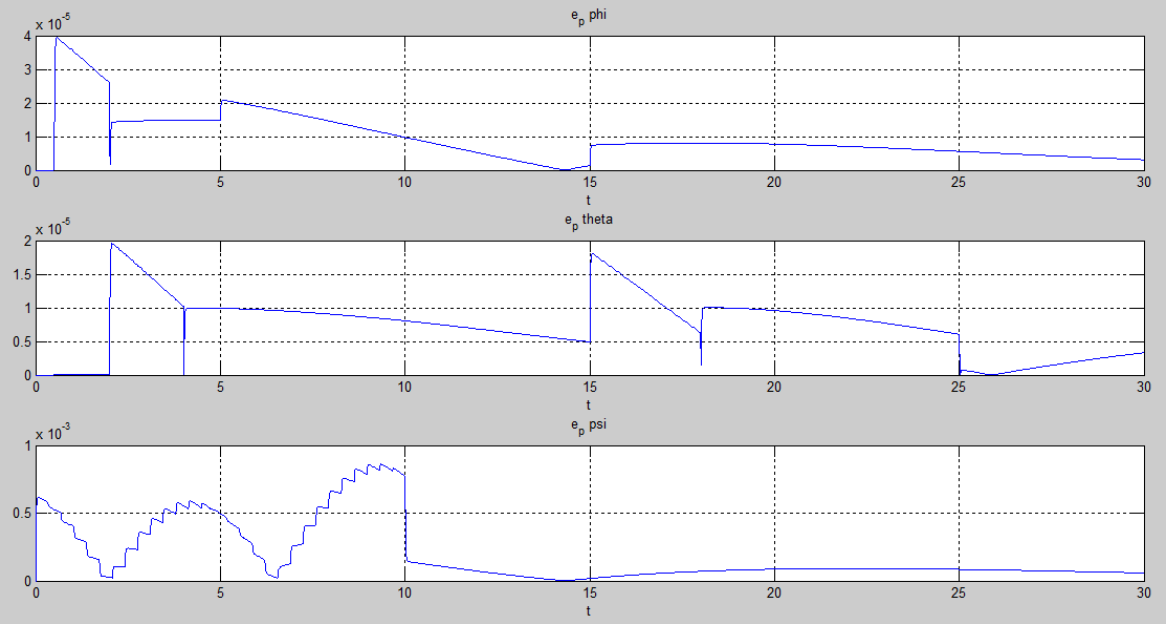



Figure 3.3-16 Position error for the nonlinear controller only

Once the inner controller works properly, the next step is to tune the values of the outer controller. The entries are the same as in the previous controller, a helicoidally trajectory. The trajectory generator outputs the desired trajectory with its derivation and integration, and the desired angles, thrust, $u_x$ and $u_y$ for this trajectory if it is done by a virtual vehicle.

The initial condition of the angles is not known a priori so a big peak appears at the first instants, as it can be seen in fFigure 3.3-17. If this peak is not eliminated, the global system gives some problems when it is simulated.

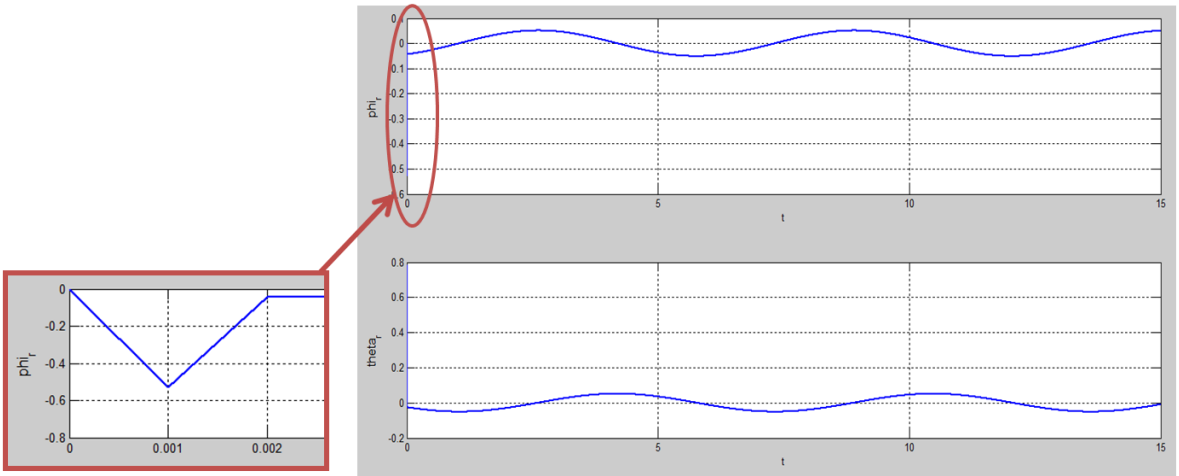



Figure 3.3-17 Desired angles coming out of the trajectory generator

This problem is solved by changing the first value of these variables to zero in the "allocation of the variables" matlab file.

Choosing the right values for this controller is quite difficult. Moreover, these values depend so much on the entry: they might be completely different when the entry changes. The values that have been selected for the helicoidally trajectory are:

| x | | | | y | | | | z | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Qx1 | Qx2 | Qx3 | Rx | Qy1 | Qy2 | Qy3 | Ry | Qz1 | Qz2 | Qz3 | Rz |
| 2100 | 0,3 | 0,5 | 0,5 | 60 | 0,3 | 0,01 | 1 | 100 | 0,5 | 100 | 0,00005 |

Table 8: Model predictive controller selected constants

For the constants above, the following output is obtained.

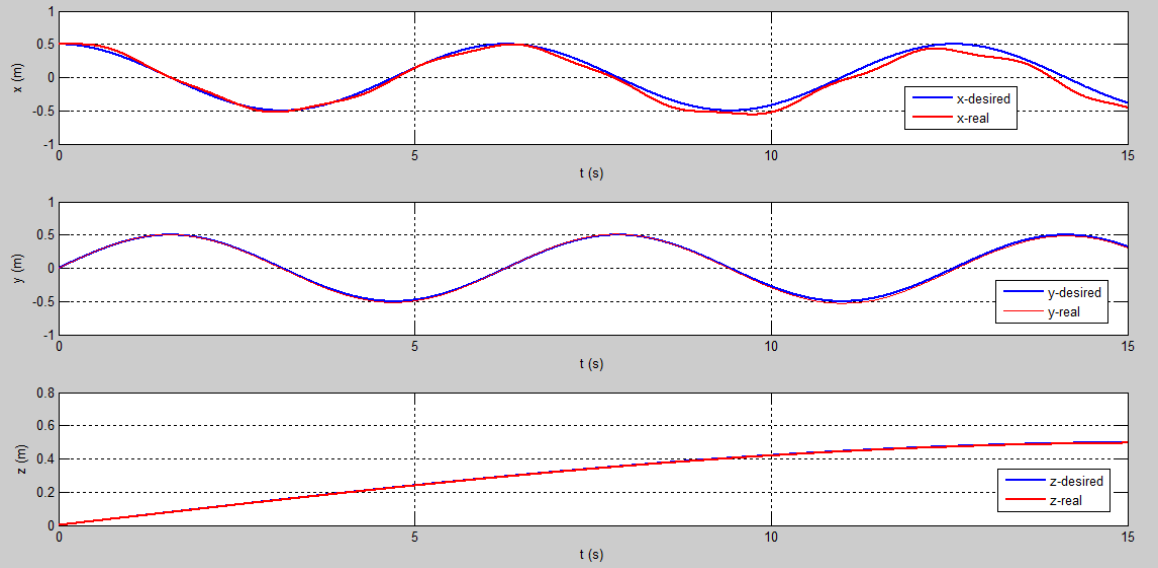



Figure 3.3-18 Desired and current position in 3 axes with the integral predictive/nonlinear controller

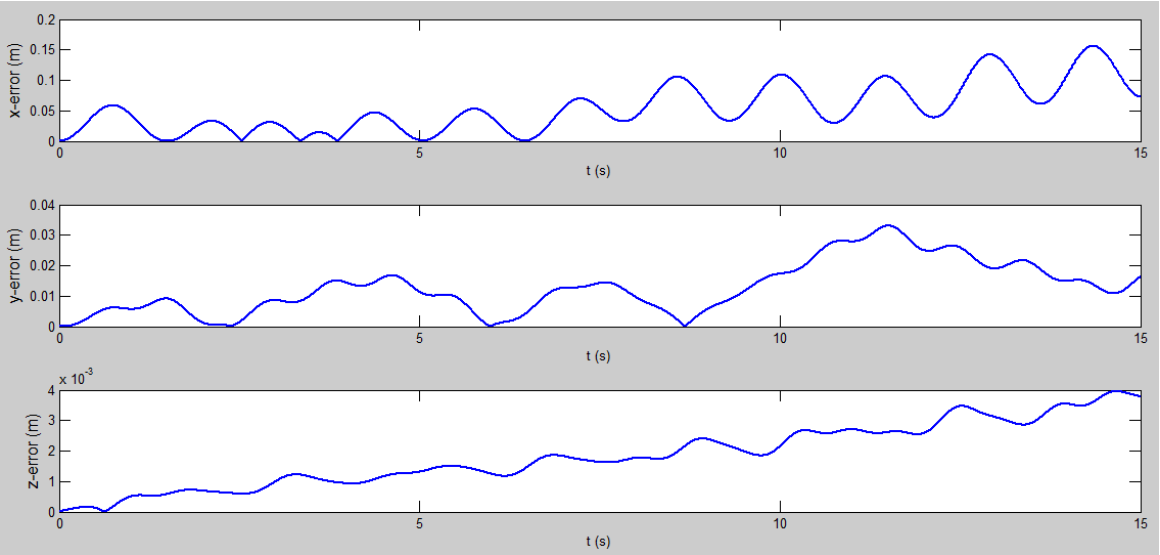



Figure 3.3-19 Position errors in 3 axes with the integral predictive/nonlinear controller

It can be seen that coordinates y and z are followed in an accurate way. The first one has an error on the order of centimeters and the second one on millimeters. However, the error in the x axis is bigger; it has a maximum value of 15 centimeters. A lot of efforts have been done in order to find a better behavior for x. Nevertheless, when this value was improved, the others increase considerably. Finally, this option was elected as the most appropriate. In the following picture, the total thrust and torque are shown.

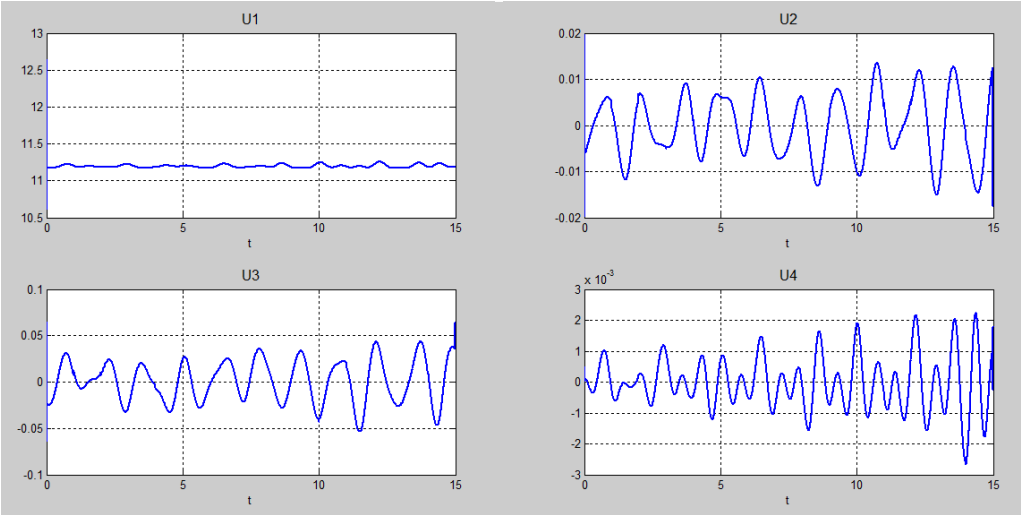



Figure 3.3-20 Total thrust and torques for an helicoidally entry with the integral predictive/nonlinear controller

In Figure 3.3-19, it can be seen that the error lightly augments with time. This system is not stable. In fact, if the previous system is simulated for a larger period of time there is a time when the actual position no longer represents the desired one.

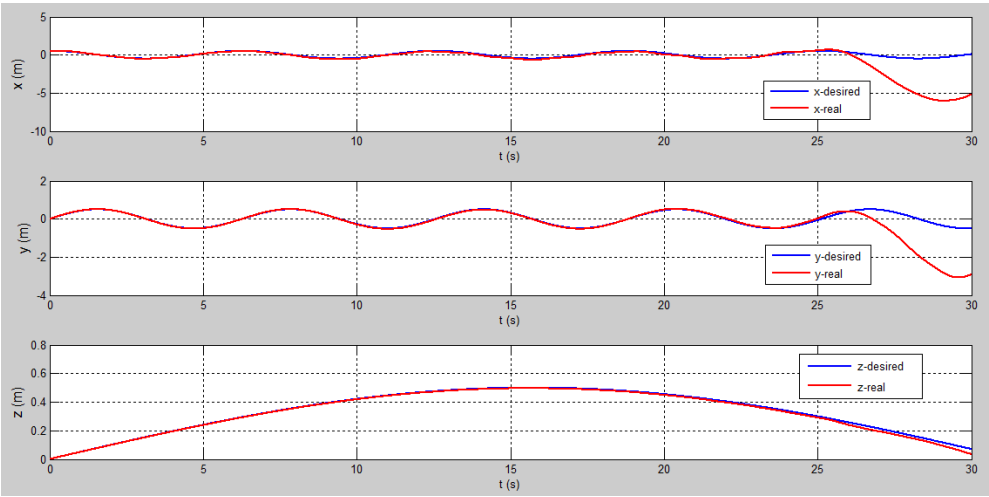



Figure 3.3-21 Current and desired position for a longer period of time

As it has been said, this controller is very sensible in the tuning of the constants. In the following picture, it can be seen how a small change in the variable $R_y$ from 1 to 0.75 has a huge impact in the response.

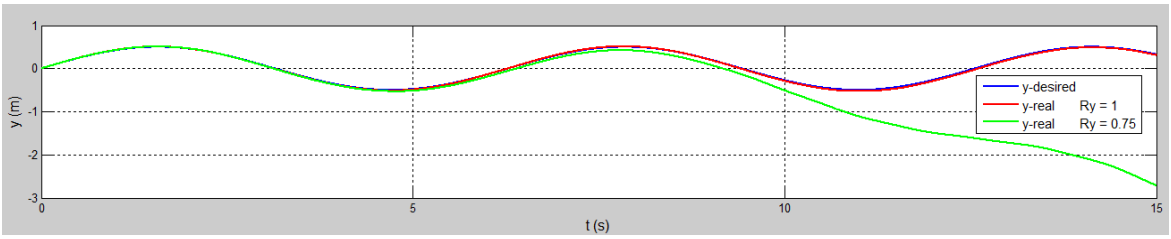



Figure 3.3-22 Current position changing the constants

To conclude, the results of this controller are going to be compared with the ones obtained in the IB. It can be observed that the IB has so much better results. However, both of them are capable to reproduce the desired trajectory. Maybe, using numeric methods, the selection of the constants of the MPC could be improved and the last controller could overcome the IB, here it has not being achieved.

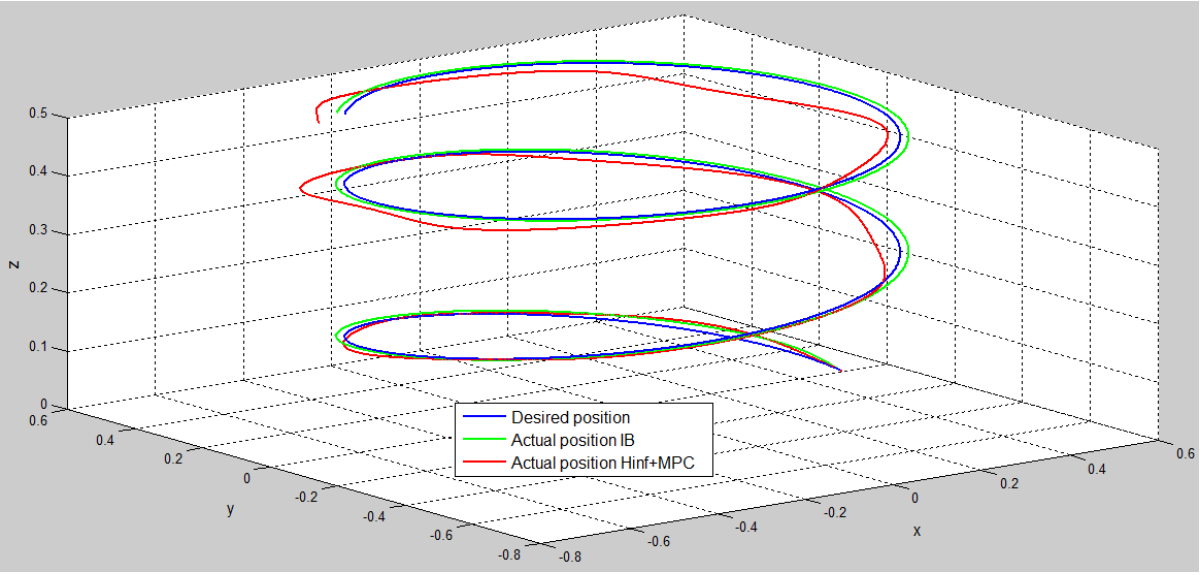



Figure 3.3-23 Comparison between current positions with the IB and the $H_\infty$ MPC

## 3.4 Trajectory generator

During section 3, the desired trajectory of the quadcopter for a period of time has been introduced using the block "Signal Builder". With this block many different kinds of signals can be created easily by adding some points. However, in some cases, it may not matter the path followed by the quadcopter but only that it reaches a certain point from an initial position. In this section, a trajectory generator block is developed. Given the three coordinates of the initial position and the final one it generates the path that the quadcopter should follow. There are multiple ways to move from one point to another in space. Depending on the situation, the best option might be the one that arrives to the desired position faster, consumes less energy, preserves better the lifetime of the unit or completes the path within a given period of time.

In (Réalisation des commandes de vol d'un drone type hélicoptère, 2012) a trajectory generator for a helicopter is developed. The method is adapted here for a quadcopter. In this work, a velocity curve will be built from the position difference. The curve will have a trapezoidal shape which means that the quadcopter will increase its speed when leaving the initial position; later it will remain with steady speed and finally reduce its speed when approaching the final point to avoid overtaking.

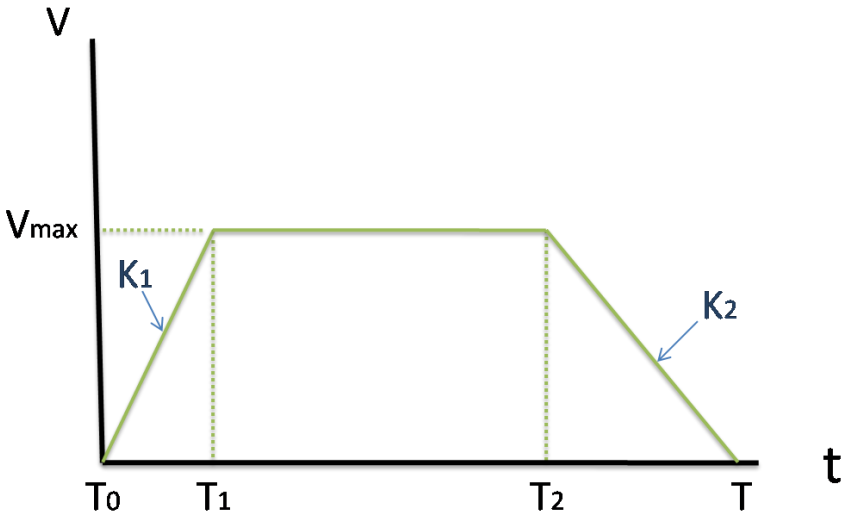



Figure 3.4-1 Desired speed curve

The mathematical model is based on that the integral of the velocity has to be equal to the position for each of the coordinates. The integration can be separated into the three different zones.

$$\delta x = \int_0^T V_x(t)dt = \int_{T_0}^{T_1} V_x(t)dt + \int_{T_1}^{T_2} V_x(t)dt + \int_{T_2}^{T} V_x(t)dt \quad (137)$$

The solution of the first and the third terms is the area of a triangle and for the other, the area of a rectangle.

$$\delta x = \frac{\left(V_{x^{max}} - V_{x_0}\right)\cdot(T - T_2)}{2} + V_{x^{max}} \cdot (T_2 - T_1) + \frac{\left(V_{x^{max}} - V_{x_f}\right)\cdot(T_1 - T_0)}{2} \quad (138)$$

Equation (138) has three unknowns that are $T_1$, $T_2$ and $T$. Assuming that the slopes are known, two more equations can be added to have a three equations and three unknowns system.

$$K_1 = \frac{V_{x^{max}} - V_{x_0}}{T_1 - T_0} \qquad K_2 = \frac{V_{x^{max}} - V_{x_f}}{T_2 - T} \quad (139)$$

In the last expressions the variables $T_1$ and $T_2$ are isolated obtaining:

$$T_1 = \frac{V_{x^{max}} - V_{x_0}}{K_1} + T_0 \qquad T_2 = \frac{V_{x^{max}} - V_{x_f}}{K_2} + T \quad (140)$$

Introducing these equations in (138) and solving for T, an equation in which all the variables are known is shown:

$$T = \frac{\delta x}{V_{x^{max}}} - \frac{\left(V_{x^{max}} - V_{x_f}\right)^2}{2K_1 V_{x^{max}}} - \frac{V_{x^{max}} - V_{x_f}}{K_2} + T_0 + \frac{V_{x^{max}} - V_{x_0}}{K_1} + \frac{\left(V_{x^{max}} - V_{x_f}\right)^2}{2K_2 V_{x^{max}}} \quad (141)$$

If T is placed in equation (140), $T_1$ and $T_2$ are also known, so the system is solved. Below, the equations of the speed curve depending on time are developed:

$$\begin{aligned} t < T_1 \;&\rightarrow\; V_x(t) = V_{x_0} + K_1 \cdot (t - T_0) \\ t \geq T_1 \;\&\; t < T_2 \;&\rightarrow\; V_x(t) = V_{x^{max}} \\ t < T \;\&\; t \geq T_2 \;&\rightarrow\; V_x(t) = V_{x_f} + K_2 \cdot (t - T) \\ t \geq T \;&\rightarrow\; V_x(t) = V_{x_f} \end{aligned} \tag{142}$$

The vehicle must not arrive to the desired position with a high velocity. For this purpose, the value $K_2$ introduced in the equation above is modified so that the acceleration approaches smoothly the desired point.

$$K_{2^{approac\ h}} = K_2 \cdot \left(1 - e^{-\alpha|t-T|}\right) \tag{143}$$

### *3.4.1 Implementation in Simulink*

A trajectory generation block has been added to the Integral Backstepping. Switch blocks have been added in each controlled coordinate. Writing 0 or 1 in the grey constant block, the system can change from automatic mode (the system calculates the trajectory based on the initial and final position) to assisted mode (the complete desired curve is drawn).

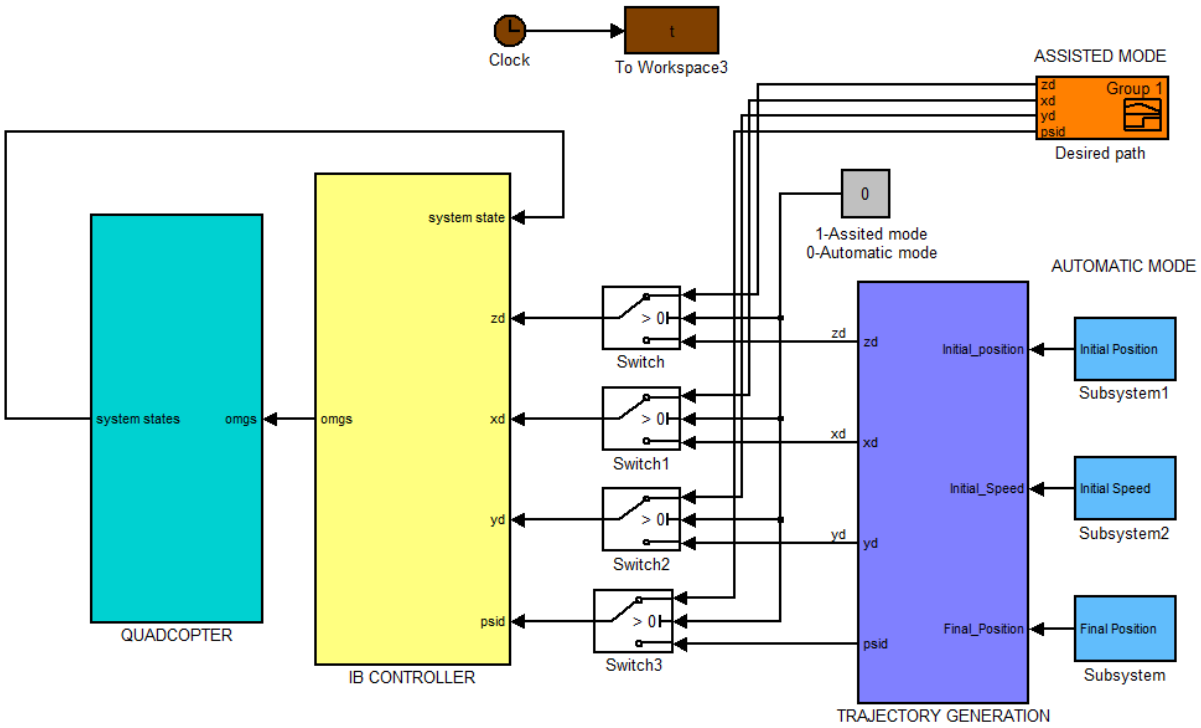



Figure 3.4-2 Trajectory Generator implementation in Simulink

In the purple block of Figure 3.4-2 there are three blocks as the one show below. The green subsystems contain the equations (140) and (141). Moreover, the value $K_{2^{approac\ h}}$ is calculated in the bottom of the graph using equation (143).

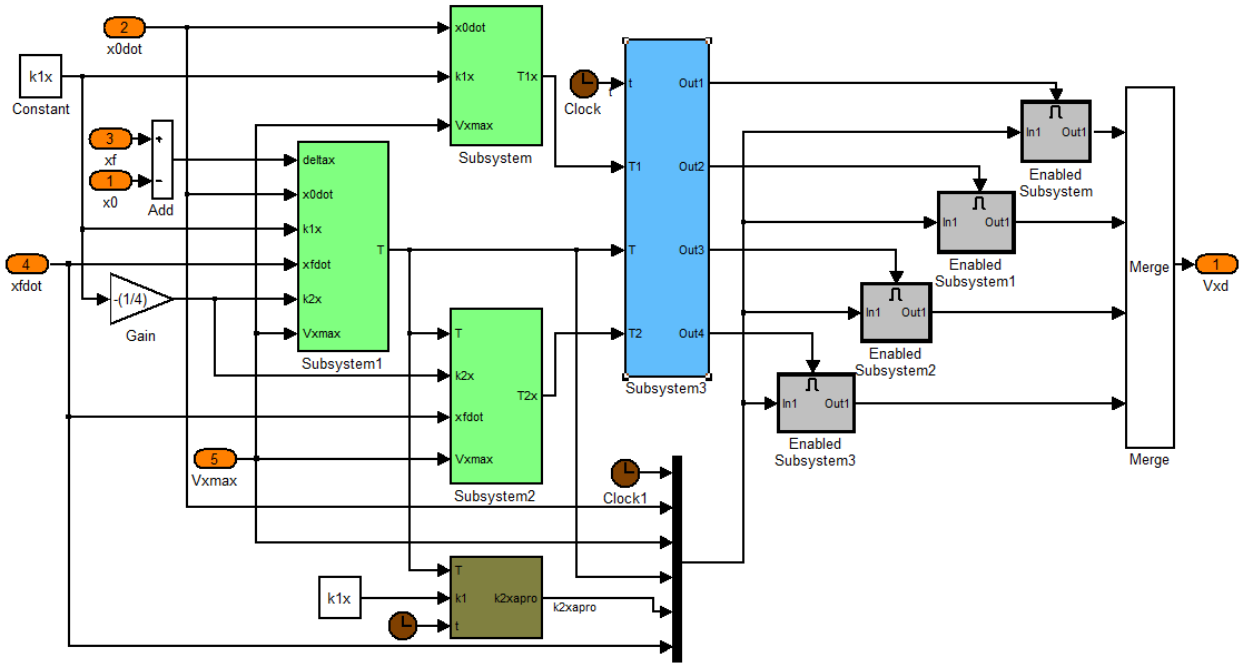



Figure 3.4-3 Trajectory Generator subsystem

The grey blocks are “enable blocks”, they include de velocity commands in (142). These blocks are only executed when the control signal has a positive value. For this purpose, the blue block, which is represented in Figure 3.4-4, generates a one in the output that corresponds to the current time instant.

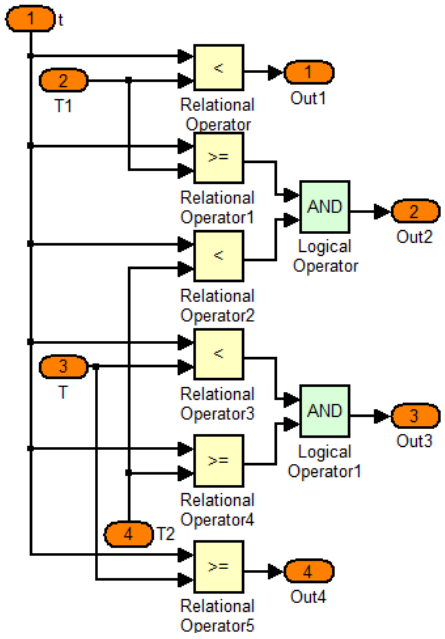



Figure 3.4-4 Relational Operator block

### 3.4.2 Simulation results

The main problem with this approach is choosing the adequate values of the slopes and the maximum velocity. They must be selecting taking into account the dynamics of the quadcopter, avoiding the saturation of the angles and the overshoot. The cruising speed that the quadcopter reaches depends on the distance between the two points: the greater the distance, the greater the speed of the quadcopter. Therefore, it can be expressed as:

$$V_{x^{max}} = K \cdot \delta x \tag{144}$$

As to the acceleration and deceleration slopes, the first will be defined and the second will be proportional to the acceleration slope but less pronounced.

$$K_2 = -C \cdot K_1 \quad \text{with} \quad C < 1 \tag{145}$$

The values that must be selected are then *K*, $K_1$ and *C*. When the value of the final position is smaller than the initial position, the sign of $K_1$ must be negative. An "if" sentence has been included in the program that holds the constants to consider this case.

However, the system defined in (140) and (141) has a problem. Once the constants have been selected, it can occur that for some cases the value $T_2$ is bigger than $T_1$. In this situation, the shape of the curve is no longer trapezoidal and the vehicle fails to reach the desired final position. This means that even if the selective constants are adequate in terms of dynamics, they must be a certain relationship between them. Supposing that the initial and final velocities are zero, as well as the initial time the inequality that must be satisfied:

$$T_2 > T_1$$

Introducing the values in (140) it is obtained:

$$\frac{\delta x}{V_{x^{max}}} - \frac{V_{x^{max}}}{2K_1} + \frac{V_{x^{max}}}{2K_2} > 0$$

Substituting $V_{x^{max}}$ and $K_2$ :

$$\frac{\delta x}{K \cdot \delta x} - \frac{K \cdot \delta x}{2K_1} - \frac{K \cdot \delta x}{2CK_1} > 0$$

and isolating $\delta x$ :

$$\delta x_{max} = \frac{2K_1}{K^2\left(1 + \frac{1}{C}\right)} \tag{146}$$

From the above equation is deduced that the range for which the equations are valid increases if acceleration slope and C are increased and the constant $K$ is decreased. To solve this problem, the path can be divided into segments that are smaller than $\delta x_{max}$ . It will be various

final points, when the quadcopter arrives at the first one; the system is simulated again having as initial point the previous final point. The process is repeated until the last end position is reached.

The selected values of the constants are shown in the following chart:

| | $K_1$ | C | K |
|---|---|---|---|
| x | 0,5 | 0,25 | 0,18 |
| y | 0,5 | 0,25 | 0,18 |
| z | 2 | 0,5 | 0,2 |

Table 9: Trajectory generation selected constants

It has also been introduced $\alpha = 10$.

The initial position has been set to (0,0,0) and the final position is (1,5,10). The results are shown in the following pictures. Figure 3.4-5 shows the desired trajectory (in blue) coming out of the trajectory generator. This would be the optimal trajectory since it doesn't have any overshoot; however, in the real trajectory there is a bit overshoot as it can be seen in the x graph. In the three cases, the desired final position is reach with zero steady state error.

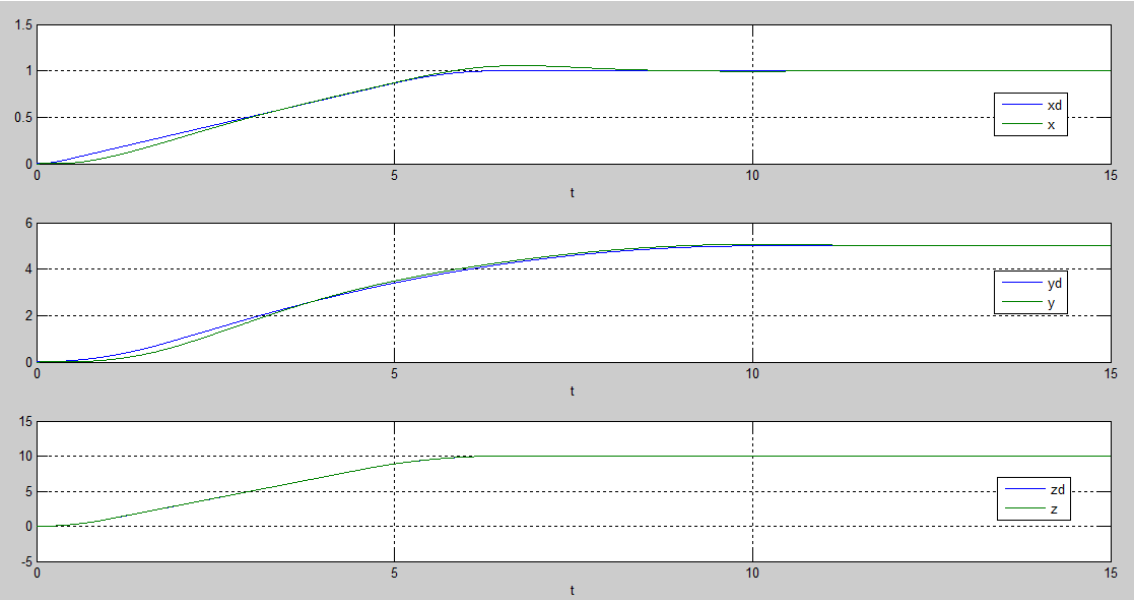



Figure 3.4-5 Desired and current position when using a trajectory generator

The velocity curves created and the ones followed by the quadcopter are shown below.

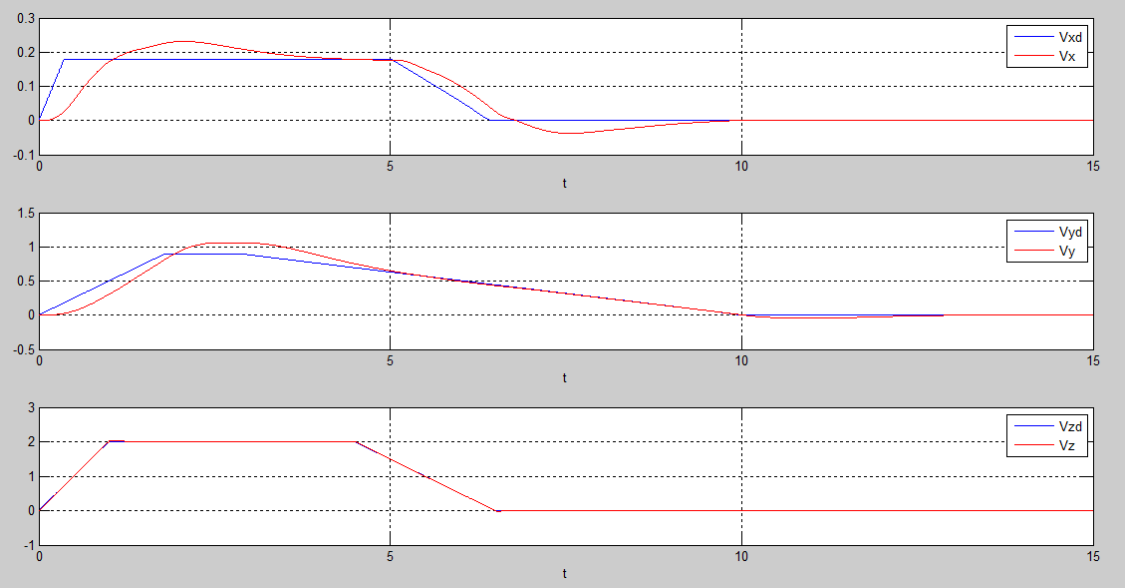



Figure 3.4-6 Desired and current speed curves when using the trajectory generator

In the z axis, the quadcopter is able to follow the created curve almost perfectly. This is due to the fact that the IB works so much better in the z axis than in the others. Another conclusion that can be extracted from this graph is that the constants selected are well chosen when the $\delta x$ is five (y position) but for smaller values (as in x) it might be other constants that are a better choice. The following picture shows that the path is followed smoothly and without exceeding the saturation limits of the angles.

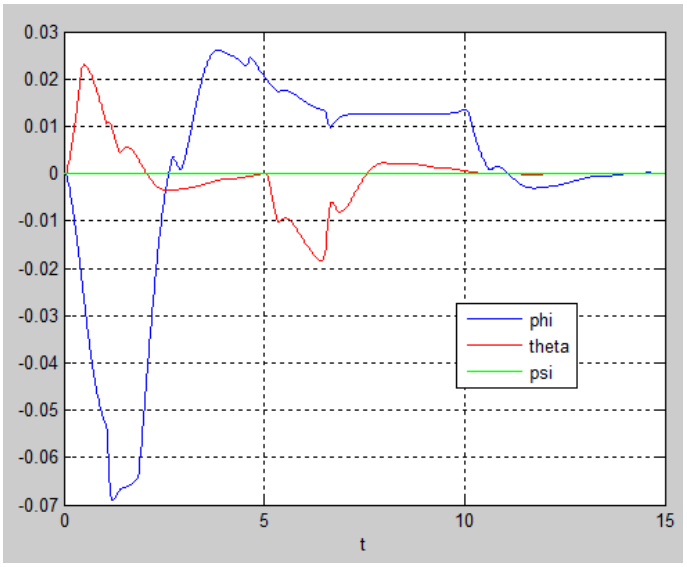



Figure 3.4-7 Attitude angles when using the trajectory generator

Introducing all the constants in equation (146) the value of $\delta x_{max}$ in this case can be known. The maximum values between initial and final positions are 6,17 in the x and y axis and 33,3 in z. The results for a $\delta x$ equal to ten in the x axis are represented in the following pictures:

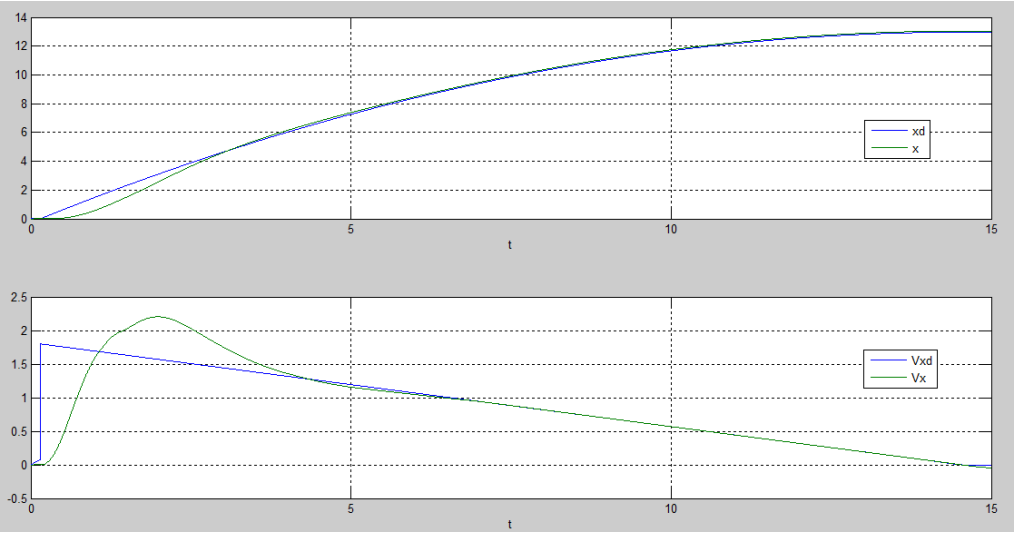



Figure 3.4-8 Position and velocity results when $\delta x_{max}$ is exceeded

As it can be noticed, the final point that is reached is not the desired one (ten) but approximately thirteen.

The results obtained for the previous constants might seem slow since the vehicle employs almost 10 seconds to move 5 meters. Moreover, the angles are not even closed to their saturation values which means that more speed can be achieved. The new values of the constants are shown below.

| | $K_1$ | C | K |
|---|---|---|---|
| x | 2 | 0,5 | 0,2 |
| y | 2 | 0,5 | 0,2 |
| z | 4 | 0,5 | 0,5 |

Table 10: New trajectory generator selected constants

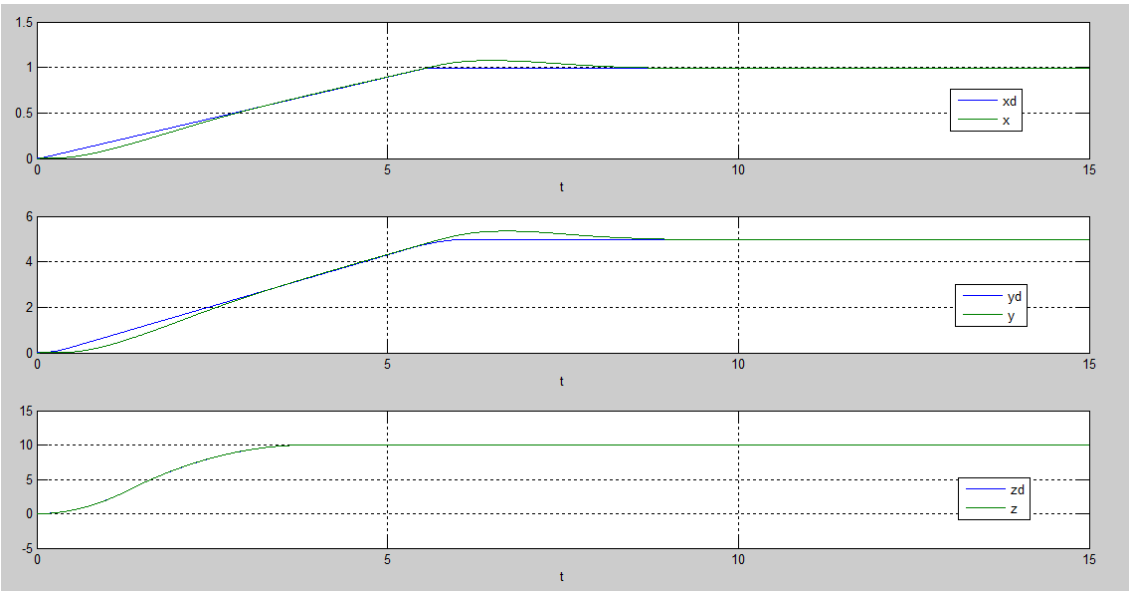



Figure 3.4-9 Desired and current position when using a trajectory generator with new constants

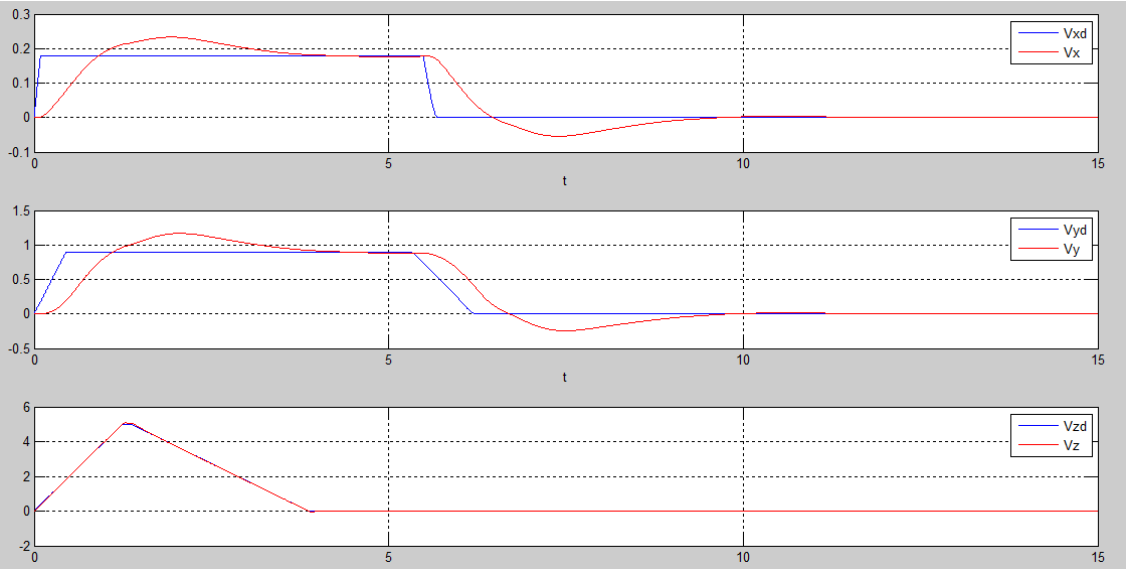



Figure 3.4-10 Desired and current speed curves when using the trajectory generator with new constants

As it can be noticed, the quadcopter arrives sooner to its destiny. With other values the speed could have been even bigger, but the overshoot increases too.

# Chapter 4

# Summary and conclusions

This study has dealt with the development of different control strategies to solve the path tracking problem for autonomous aerial vehicles, in this case, a quadrotor helicopter on a small scale. The characteristics of these vehicles such as maneuverability, hovering or vertical and indoor flight make them very useful in tasks like aerial photography, supervision or monitoring, recognition of disasters or precision agriculture. As research continues to push the boundaries of quadrotor capabilities, these and many more applications will unfold.

In **Chapter 2**, initially the principles of the quadcopter flight have been described. This enhances the knowledge of dynamics in 3D space. It has been seen how the movement of the vehicle depends only on the difference in rotational speed of the four rotors, it cannot move translational without first rotate around one of its axes. A dynamic model of the system must be obtained to design control strategies, taking in mind a tradeoff between complexity and realism. The quadrotor aircraft is a highly non-linear, Multi-Input Multi-Output (MIMO), strongly coupled and underactuated system with only four actuators. Some assumptions have been made in order to simplify the dynamics of that complex system to be suitable for simulation. The assumptions have considered the center of mass and the center of rotation as congruent; also, the aerodynamic effects are neglected, thrust and drag force are modeled as proportional to the square of the motor speed and the motors; the structure is supposed to be rigid and the ground effect is neglected.

Obtaining the equations of motion requires knowledge of rotation matrices which converts forces between different reference systems. This conversion is also described in the report. The forces and torques generated by the propeller's thrust are expressed. It can be observed that they only depend on the rotational speed of the rotors and three constants. Gravity and gyroscopic effects are also taken into account and expressed in adequate axes. The dynamic equations of the helicopter based on two approaches have been computed. Firstly, the method of classical mechanics has been adopted which is based on Newton's Laws. After that, the Lagrange Method has been described. In both methods, the small angles assumption must be done in order to simplify the equations. This may seem a very restrictive measure; however, even with a very small tilt, the quadcopter is able to move faster to the desired horizontal direction. Therefore, high-speed movements are not extremely limited by this action. It has been seen that the motors used are small and electric, their inertia is therefore very little and the response almost instantaneous. This reason, coupled with the lack of data on their behavior, has caused the non-inclusion in the system. The system equations of the model show that the angles dynamics is independent and it does not rely on translational dynamics while translational dynamics are dependent on the angular dynamics, this is clear from the

equations of motion of translation as they are function of ϕ, θ, and $\psi$. The mathematical model obtained, has been modeled and simulated using Simulink program. Hovering conditions have been calculated and thereafter it has been successfully checked that the model reacts as predicted.

Finally, a simplified model of the laboratory's quadcopter has been designed in Catia, the masses of each part have been introduced in the model which automatically gives the inertia moments. Another option could have been to calculate the moments mathematically decomposing the structure in prisms and cylinders. However, the results obtained with Catia were comparable to the ones conducted in other works with similar quadcopters. The thrust and drag constant are two values of great importance in the dynamics of the quadcopter. They are directly responsible for relating the speed of the motors with the force and moments generated. In this study these values have been calculated roughly, comparing them with other works. For control purposes, it is admissible; however, if it is planned to incorporate the control systems in the real quadcopter this values should be calculated more accurately. After that, the controller would need to be readjusted.

In **Chapter 3**, some different control techniques have been used to control both rotational and translational movements. The whole control algorithm is used to give the right signals to the propellers. Since they are four, no more than four variables can be controlled in the loop. For the first controller, the **PID**, it has been decided to stabilize attitude (Euler angles) and height. Not being able to control the x and y position is not a problem if the quadcopter is being piloted remotely and it is in sight. The pilot would tilt the vehicle by changing gently the angles ϕ, θ or $\psi$ to direct it to the desired position. It has been tried to calculate the constants of this controller by the Ziegler-Nichols procedure. The results have not been satisfactory for the quadcopter's requirements. However, from these results and knowing how the value of the constants affects the system, adequate constants have been found. Sinusoidal and ramp commands are perfectly reproduced by the vehicle. Regarding step inputs, they are reproduced with almost zero overshoot and steady-state error and an acceptable rise time. Some random noise has been added to the system simulating one of the measurement instruments; also, huge rapid changes in attitude angles simulating wind gust have been included. In the last case, the quadcopter leaves the desired path but recovers it soon.

A good controller should be able to reach a desired Cartesian position (x, y, z) and a desired yaw angle ($\psi$) while keeping the stabilization of the pitch (θ) and roll (ϕ) angles. This is achieved by the Integral Backstepping controller. It is more complex than the previous one and more computation is required. However, convergence of the states is guaranteed as it has been proven through different inputs. After adjusting all the constants, following a similar execution as in the previous case the results were very accurate. Different types of reference trajectories were simulated obtaining little overshoot and zero steady-state error. This last characteristic is achieved by the integral term added to the backstepping. One Newton forces were introduced in each one of the axes at different times. The helicoidally trajectory that was simulated is slightly disturbed especially when the perturbation is in the z axis, however, the vehicle gets back to the reference trajectory.

For the last controller, an **MPC** algorithm has been included in Simulink, as well as a **nonlinear $H_\infty$** controller for the helicopter stabilization in the inner loop. The integral of the position error has also been considered for this controller, in order to reject disturbances in the System. The tunability of the constants in this case has been more difficult than in the other cases. The outputs of the system were not stable; even if at the beginning of the simulations the results were good, they tend to infinity. Finally, some constants that reproduced the quadcopter path were found. However, the error in x is still important and when it was tried to be reduced the y error increased. Theoretically this controller should be able to reject disturbances due to the inclusion of the integral term. However, the results in this direction have not been satisfactory. The time the perturbation is introduced, the output goes to infinity. The main reason to use the MPC is due to its predictive features. Since the reference trajectory is usually known, the MPC can provide a smooth path tracking. This characteristic has been achieved in the simulation. Comparing the values of U1, U2, U3 and U4 with those in the IB, it can be observed that variations are much less abrupt.

To conclude, a trajectory generator has been developed. Given an initial position the vehicle is capable of reaching the final desired position defining by itself a trajectory. This trajectory has a trapezoidal shape in velocities. Three constants have been adjusted in other to define the shape: the acceleration constant, the relation between the acceleration and deceleration constants, and the one that relates maximum velocity of the curve with the distance from the initial to the final point. Two groups of constants have been simulated with the IB controller. The second one reaches sooner the objective. The deceleration slope has been modified so the arrival is produced without overshoot. The only problem with this approach is that the distance between origin and destiny cannot be as big as the desired due to the definition of the equations.

## 4.1 Possible future work

A general approach to the embedded system of the quadcopter has been carried out in another internship. It would be very interesting to be able to integrate the developed control systems into the embedded system and modify them directly with Simulink. The software Gene-Auto converts the simulation models into C code. Since Gene-Auto only admits certain Simulink blocks, the models must be modify. The integral backstepping and MPC are complex and the conversion in C would take quite some time, although it would be an interesting work. The PID controller has been converted into C code, as it can be seen in (Đạt, 2014) but it has not been introduced in the embedded system jet.

Some assumptions have been done throughout this study. Therefore, some test must be done in order to define accurately the variables prior to flying the helicopter. Once the new variables are calculated, the constants of the PID should be redefined with Simulink. Finally, the last readjusting should be done in flight.

Regarding controllers developed in the project, some tasks could be done:

- Mathematically optimization of the constants depending on the output characteristics desired: overshoot, steady-state error, rise time…
- The control signals that come out of the IB controller are very abrupt. Since these are the values that would be introduced into the embedded system of the quadcopter, it would be recommended to use a filter that smoothes the signal.
- The last controller's behavior is not as good rejecting perturbations as it is shown in the bibliography. New approaches could be done in order to achieve this characteristic.
- There are different ways to realize a path generator. The one carried out in this study was satisfactory. However, it has some problems with the magnitude of the distance to cover. New focuses could be developed to ensure good performances regardless of the distance.

# Bibliography


*3D Robotics*. http://www.3drobotics.com/iris/

Abolghasem Naghash, Mehrdad Naghshineh and Ali Honari. (2013). Minimum Time Trajectory Optimization for Flying a Quadrotor in an 8-shaped Path. *International Micro Air Vehicle Conference and Flight Competition (IMAV2013)*, (pág. 4). Toulouse.

Agarwal, S. (2012). *Monocular Vision Based Indoor Simultaneous.* Cranfield University. Bedfordshire, UK.

*Airbus Defence and Space*. http://www.cassidian.com/es_ES/web/guest/barracuda

Ariza, L. M. (13-02-2014). Drones. *El Pais* .

*BAE Systems*. http://www.baesystems.com/Taranis

Basta, P. O. (2012). *Quad Copter Flight.* California State University, Northridge.

BOUABDAHALL, S. (2007). *Design and control of quadrotors with application to autonomous flying.* Thesis , Lausanne, EPFL.

Crenne, B. (2012). *Réalisation des commandes de vol d'un drone type hélicoptère.* Poitiers.

Đạt, B. N. (2014). *Embedded System for Quadcopter.* Poitiers.

*Federal Aviation Administration*. http://www.faa.gov/about/initiatives/uas/

G. V. Raffo, M. G. Ortega, and F. R. Rubio. (2010c). An integral predictive/nonlinear Hinf. *Automatica, 46:29–39* .

*General Atomics Aeronautical*. http://www.ga-asi.com/products/aircraft/index.php

Hoffmann, G. M. *Quadrotor Helicopter Flight Dynamics and Control.* American Institute of Aeronautics and Astronautics.

*Israel Aerospace Industries*. http://www.iai.co.il/2013/18900-en/IAI.aspx

K.Alexis, G. Nikolakopoulos and A. Tzes. (2011). Switching model predictive attitude control for a quadrotor helicopter. *Elsevier* .

Miguel Angel Gómez Tierno; Manuel Pérez Cortés. (2012). *Mecánica de vuelo.* Madrid: GARCETA GRUPO EDITORIAL.

*Northrop Grumman*. http://www.northropgrumman.com/capabilities/globalhawk/Pages/default.aspx

*Parrot*. http://www.parrot.com/productos/

Raffo, G. V. (2011). *Robust control strategies for a Quadrotor helicopter. An Underactuated Mechanical System.* Sevilla.

(2013). *Roadmap for the integration of civil RPAS into the European Aviation System.*

Yanmin Chen, Yongling He and Minfeng Zhou. (2013). *Modeling and Control of a Quadrotor Helicopter System under Impact of Wind Field.* Beijing.