Controlling a quad-rotor with a robotic arm using a NMPC

Seabstián Chávez-Ferrer Marcos

Institut de Robòtica i Informàtica Industrial

Universitat Politècnica de Catalunya

A thesis submitted for the master of

Automatic Control and Robotics

Acknowledgements

I would like to express my acknowledgements to the Master coordinator, Cecilio Angulo, who helps me to solve all my administrative problems and also who showed me the benefits of the master.

Also, I would like to thank to my supervisor Carlos Ocampo, who guided me through the goods and bads moments of the thesis. Without its help it would have been impossible to finish it and get these results.

On the other hand, I would like to express my appreciation to all the people from the Institut de Robòtica i Informàtica Industrial (IRI) where I was very well welcomed and where was helped me to solve most of my technical problems.

I would like to thank to all the people who suffered and enjoyed this Master with me, my class colleagues from the MAiR and my friends from ETSEIB.

Finally I would like to express my biggest acknowledgements to my parents, who supported me and encouraged me until the end of my studies.

Abstract

This thesis develops a method to apply the control to a quadrotor equipped with a robotic arm developed in Institut de Robòtica i Informàtica Industrial (IRI).

During the project, an algorithm has been built as a first approximation to control a quadrotor which is working with a robotic arm.

In order to compensate the perturbations of the arm's dynamic, a nonlinear model predictive control algorithm has been chosen among other possible techniques.

Finally, several scenarios has been simulated and tested to verify the performances and robustness of the method created.

Contents

List of Figures

List of Tables

Nomenclature

A	Rotor disc area
C_Q	Non-dimensional torque coefficient
C_T	Non-dimensional thrust coefficient
CoG	Center of gravity
D_i	Rotor displacement from the flyer center of mass
I	Rotational inertia
IRI	Institut de Robòtica i Informàtica Industrial
ISE	Integral of the Square of the Error
I_b	Rotor blade rotational inertia about the flapping hinge
M_i	Momentum due the displacement of the thrust relative to the center of gravity
Q_i	Torque from the i-th rotor
R	Rotation matrix
T_i	Thrust from the i-th rotor
ω_i	Angular velocity of rotor
ρ	Density of the air
σ	Rotor solidity is the ratio of the total blade area to the total disk area

a	B
$a1s_i$	Non-dimensional longitudinal flapping coefficient
$b1s_i$	Non-dimensional lateral flapping coefficient
c	Blade chord
g	Acceleration due the gravity
m	Mass of the Rotor
r	R
r	Rotor radius
$sk(\Omega)$	Skew-Symmetric matrix
t	Blade tip angle
UAV	Unmanned Aerial Vehicles

Chapter 1

Introduction

This project has been motivated by the need of extend the capabilities of a quadrotor. Most of the literatures explain methods to control an UAV applying MPC to a linearized model.[3]

Other kind of papers are focused to find a properly mathematical model that explains most of the dynamics effects that occurs during the maneuvers.[12]

The Institut de Robòtica i Informàtica Industrial (IRI) is designing a quadrotor equipped with a robotic arm. This project is developed in an european framework called ARCAS (Aerial Robotics Cooperative Assembly System). [19] The goal of this research is produce a framework for the design and development of cooperating flying robots for assembly operations.

The thesis presented here has the aim of develop a first approach of controller able to manage the perturbations due the equipped arm. So an algorithm has been developed to study the viability of the control.

Chapter 2

Background information and Theory

2.1 State of the art

In the recent years, the relevance of the Unmanned Aerial Vehicles (UAV) has been increased drastically. The integration of the UAV in our society will be completed in the next few years.

It is possible to delegate to an UAV a lots of tasks that for a human would take much more time and risk.

Nowadays there are few real examples where the UAV could be really helpful, and most of them are in the military field. On the other hand, most of those need the human intervention to achieve its goal. For example aerial photography, television and cinema shootings or research which need flaying to perform aerial experiments.

However, exist a huge number of potential applications where the UAV is completely autonomous that still could be developed. Actually most of the big companies are working to implement new services using autonomous UAV. Surveillance, crowd control, aerial delivery of payload or mine detection could be some of the easiest examples.

But in order to accomplish it, is necessary to develop new methods and algorithms to control these vehicles in a non controlled environment. Right now

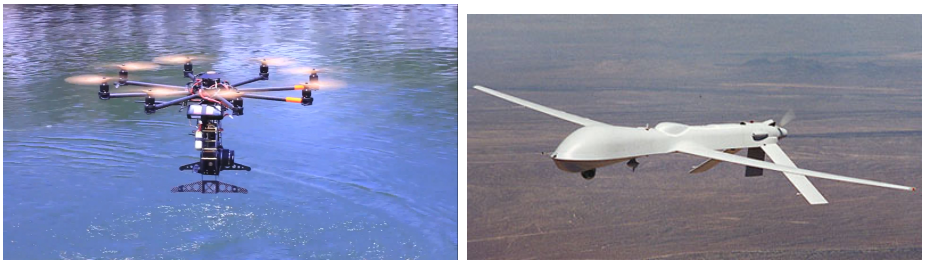

Figure 2.1: Examples of UAV

almost all the he external perturbations are compensated by a human. But the final target is find a method able to understand the situation, to predict what is going to happen and to correct the behaviour of the UAV in order to accomplish the performance necessary to reach the goal.

At present, the most popular UAV integrated in our society is the Quadrotor. This mechanism was concepted in 1907 by Breguet and Richet. The first model was a large and heavy model that could lifted only over a small height and for short duration. At present, and after a lot of efforts the quadrotor has more attention and each day has more relevance for the companies.

On the other hand, the use of the robotics arms is widespread in all areas. These tools are helpful to accomplish several tasks. Also, its dynamics and behaviors are well known. There are a lot of studies and papers about how to control and minimize the error so it is a good choice to use it for add new features to a quadrotor. New applications could be developed like collaborative tasks to move objects by pincers, track object with a camera in the end effector or anchor the quadrotors to recharge its batteries.

During this thesis, a method to control a quadrotor with a robotic arm is presented. The aim of this work is explore the capabilities of an UAV to compensate the arm's dynamics during different movement phases of the quadrotor.

In this way, chapter 2 has all the information about the technology used to make this thesis and the theoretical explanation of the methods used.

In chapter 3 a description of the implementation is presented to the reader. Next, in chapter 4 the results of the methods applied will be shown with a full detailed information about the environment. Finally, in chapter 5, a conclusion about the thesis and a future work is presented for the interested readers.

2.2 Background Information

As it is said before, three main elements are used during this projects.

First of all an UAV, more specifically a Quadrotor. Right now, is the most popular vehicle in the commercial field. In other words, is the most studied system and also the cheapest model that the reader could find. Both reasons are the cause of why most of the researchers labs are focusing their studies on this robot.

The second element is a robotic arm. This mechanism increase the versatility of the quadrotor. At present there are a lof of kind of robotic arms. Considering that a quadrotor have a small payload, the arm has to be light but strong enough to carry the maximum possible weight. So, in that case, a robotic arm designed in the IRI has been chosen to test the methods explained in this thesis.

Finally, and the last component is the control algorithm. Among the huge number of possibilities a nonlinear model predictive control has been selected to control the whole system. The main reason is because most of the problems that have been had to face could be represented as an optimization problem. Also, because the dynamic systems of the quadrotor and the arm are well known, it is possible to translate it as a mathematical system to the model of the controller. On the other hand, due the nonlinearities, is important use a robust algorithm to absorb all the non controlled changes.

During the next sections, a more deep comparison has been done to compare the different possibilities that exists nowadays.

2.3 UAV

The UAV is an acronym for Unmanned Aerial Vehicle, which is an aircraft with no pilot on board. Usually, UAV's are remote controlled , can fly autonomously based on pre-programmed flight or using a more complex dynamics. The UAV concept is becoming more popular each year. There are a lot of types, size and prices. For the purpose of the thesis, a system with the hover capability is needed in order to hold a fixed position while an arm is working in some task. Keeping this in mind, the most current UAV used are based on system with vertical thrust.

That kind of technology consists on a generation of vertical force to compensate the gravity component of the whole dynamic system. Also, by combining the different forces, it is possible to create forces and torques in all axis to move the robot to any position with a desired motion. In this sense, the cheapest hardware and the most reliable drones are those based on rotors. It is possible to find several types of rotor systems in function of the task that it has to do. Depending of the autonomy, the payload or maneuverability, the number of rotors could change between models.

On the other hand, other feature that could change drastically the behavior of the UAV, are the propellers. By changing its geometry, the dynamic of the system could vary the parameters named above.

The case of the study is based in a model of four rotors, henceforth named quadrotor. These vehicles use a control system and electronic sensors to stabilize the aircraft. Because the small size and agile maneuverability, these UAV can be flown indoors as well as outdoors. The quadrotors have two sets of identical rotorcraft, two of them are spinning clockwise, and two counter-clockwise in order to compensate the momentum generated. Even in the quadrotor category there are several types depending of the position of the rotors relative to its center of mass.

During the thesis a X-Type has been chosen because its dynamic model are more easy to manage than other kind of structures. This geometry consists on situate the four rotors in a cross position.

Figure 2.2: X-Type quadrotor

This research pretends to use a generic model of a quadrotor and apply a

nonlinear control to accomplish certain targets. During the next chapters, a more detailed explanation about the dynamics of the model used is going to be described. Also, all the simplification done until the last model used has been justified and detailed.

2.4 Robotic arm

There are several kind of arms and a huge number of studies about its dynamics. Implement a robotic arm to a quadrotor dynamics will extend the capabilities and services that a UAV could offer.

Looking all the possibilities that exists right now and keeping in mind the constraints due the quadrotor, it is necessary use a light robotic arm.

During this thesis, other research group in the IRI has been working on design a robotic arm able to be added to a quadrotor.

This arm are light and enough strong to carry several tools on its end effector. Actually, this group is working on use a camera to track a tag using the quadrotor.

In order to control the whole systems, it is necessary to know precisely all the parameters of the robot to build a mathematical model. Usually, it is hard to find all the parameters of a commercial arm, so this is one of the reason because this design has been chosen for this thesis.

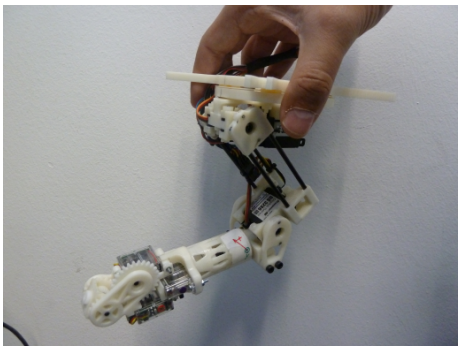

Figure 2.3: Robotic arm designed by IRI

One of the targets of this thesis is to check the viability of use the arm designed by IRI in a control based on a dynamic model.

In next chapters a full description of the arm and its dynamics it is going to be explained.

2.5 Algorithm Control

Finally, the last element necessary to control a quadrotor with an arm, it is its control algorithm.

At the beginning, a black box model was selected as the dynamic model of the whole system because the first try was study the problem as much generic as possible.

After realize that there are a strong coupled effects between quadrotor and arm, a different strategy was chosen. Those effects did not allow to find a logical relation between quadrotor and arm inputs, and the behavior of the whole system.

Due the dynamic model of a quadrotor and an arm is well known, a black box model was discarded. Once the kind of model has been chosen, the whole problem has been solved as an optimization problem.

At first, without having into account the arm, the most used method consists in linearizing the model around a certain state of the quadrotor and then applying a model predictive control around this point. This method ignores most of the dynamics effects of the system and also cannot represent accurately the real behavior of the model. Also, with this strategy, to add a new model or perturbation would be difficult.

Because of this, a nonlinear predictive model has been used during this thesis. Adding the nonlinear part improves the response of the system at the expense of increase the necessary computational resources.

On the other hand, the robotic arm has been treated as a perturbation of the quadrotor so there are no need to control its joints. But it is necessary to know the exact model of the arm to know the dynamic reactions applied to the quadrotor due to the motion of the arm during its trajectory.

The trajectory of the arm is precalculated, so it is a parameter of the system. This means that for each instant it is possible to know the reaction of the arm into the quadrotor. Because the aim of the thesis is to compensate this perturbation, a prediction algorithm is used to minimize the error.

The main problem of the strategy is that a good estimation of the state variables of the quadrotor is necessary; and it doesn't have a trivial solution.

During the next chapters, a more detailed specification of the problem has been written for the reader.

Chapter 3

Implementation

In this section, a full description of the three elements commented above has been written.

3.1 Quadrotor description

3.1.1 Mathematical model of the quadrotor

3.1.1.1 Full Model

The mathematical model based in this thesis comes from the description done in Modelling and Control of a Quad-Rotor Robot POUNDS et al. [2006]

In order to have a more accurate model of the quadrotor, it has added the flapping effects due the geometry of the propellers.

Let's suppose two frames:

1. Inertial frame: Right-Hand frame denoted by $I = \{E_x, E_y, E_z\}$ where E_z is int hte direction of gravity.
2. Body frame: A frame located in the body of the quadrotor. Its axis are denoted by $I = \{E_1^a, E_2^a, E_3^a\}$ and with the center in $\xi = \{E_x, E_y, E_z\}$.

Both frames are related by a rotation matrix $R : A \rightarrow I$

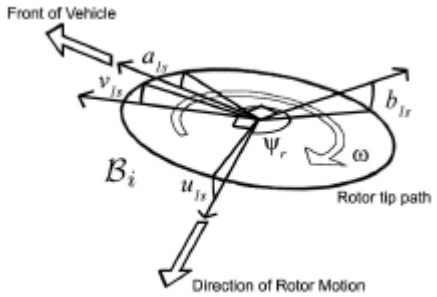

Figure 3.1: Blade system reference

Finally V and Ω are the linear and angular velocity of the frame A in base A.

The dynamics equation of the quadrotor are:

$$\dot{\xi} = R \cdot V \tag{3.1}$$

$$m \cdot \dot{V} = -m \cdot \Omega \times V + m \cdot g \cdot R^T \cdot E_3^a + \sum_1^4 T_i \tag{3.2}$$

$$\dot{\Omega} = -\Omega \cdot sk\left(\Omega\right) \tag{3.3}$$

$$I \cdot \dot{\Omega} = -\Omega \times I \cdot \Omega + \sum_1^4 \left[Q_i + M_i\right] \tag{3.4}$$

$$T_i = C_T \cdot \rho \cdot A \cdot r^2 \cdot \omega_i^2 \cdot [-cos(b1s_i) \cdot sin(a1s_i), sin(b1s_i), -cos(a1s_i) \cdot cos(b1s_i)]^T \tag{3.5}$$

$$Q_i = C_Q \cdot \rho \cdot A \cdot r^3 \cdot \omega_i \cdot |\omega_i| \cdot E_3^a \tag{3.6}$$

$$M_i = T_i \times D_i \tag{3.7}$$

Where each parameter is:

- m: Mass of the Rotor.
- I: Rotational inertia.
- ρ: Density of the air.
- g: Acceleration due the gravity.
- r: Rotor radius.
- A: Rotor disc area. This is the virtual area of the
- rotor. Depends on the geometry of the propellers.
- ω_i: Angular velocity of rotor i.
- $sk(\Omega)$:Skew-Symmetric matrix.
- R: Rotation matrix. It is constructed by the yaw-pitch-roll = (φ, θ, ψ) euler angles
- i: Rotor number i
- D_i: Is the rotor displacement from the flyer center of mass:
  - $D_1 = (0dh)$
  - $D_2 = (0 - dh)$
  - $D_3 = (d0h)$
  - $D_4 = (-d0h)$
  - where d is the arm length of the flyer and h is the height of the rotor above the CoG.
- T_i: Thrust from the i-th rotor.
- Q_i: Torque from the i-th rotor.
- M_i: Momentum due the displacement of the thrust relative to the center of gravity.

- C_T: Non-dimensional thrust coefficient. This is an experimental parameter that could vary slightly.
- C_Q: Non-dimensional torque coefficient. This is another experimental parameter.
- $a1s_i$: Non-dimensional longitudinal flapping coefficient.
- $b1s_i$: Non-dimensional lateral flapping coefficient. Both parameters are related with the blade flapping effect. This values are going to be explained a few lines later.

All this parameter, and a few more that are going to be detailed later, determine the model of the quadrotor. By changing this parameters and using the mathematical expression described above, it is possible to estimate the behavior of a quadrotor.

- 3.1 The first equation relates the linear velocity of the CoG between inertial frame and the body frame.
- 3.2 Shows the sum of the forces applied on the quadrotor have to be proportional to its linear acceleration. It is splitted in three different components. First, is the force due the coriolis effect. Second, is the force because the gravity. And finally, is the thrust generated by each rotor.
- 3.3 This expressions allows to relate the local euler angles with the euler angles of the inertial frame.
- 3.4 In the same sense, the sum of the torques applied have to be proportional to its angular acceleration. As before, the final torque it is explained in three elements. First, it is the torque created by an inertial system that is rotating around an axis. Q_i corresponds of the torque generated by each rotor due its angular velocity. M_i is the momentum created by the difference between the thrust of the rotor and the thrust of its opposite.
- 3.5 This force is generated by the angular velocity of the rotor. It is dependant of the geometry of the propellers and the density of the air. It

is multiplied by a vector that modify the final direction of the thrust. Instead of a vertical direction, appears a component on E_1^a and E_2^a due of the flapping blade effect.

- 3.6 Is the torque generated by a mass (propeller) rotating around an axis (rotor).
- 3.7 Because the thrust it is not applied in the CoG a torque appears in the frame of the quadrotor.

The quadrotor can move in the three axis by combining the thrust and the torque of each rotor:

1. Move forward-backward: To move forward or backward it is necessary to make a difference between the thrust generated by the front rotor and the back rotor. By this process the $\sum_4^1 M_i$ is not zero so the quadrotor has a slight torque that rotates the system until the forces and torques are balanced again. A pitch angle it is generated, which changes the direction of the thrust. A new component of force appears toward E_a^1 axis that allows an acceleration and finally a velocity.

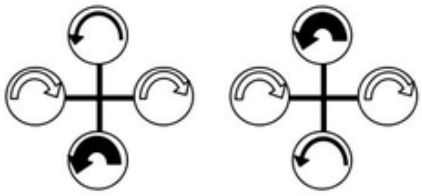

Figure 3.2: Forward-Backward movement

2. Move left-right: In order to accomplish these motions, it is necessary to create a difference between the thrust generated by the lateral rotors. With the same concept explained before, a new torque is generated and the quadrotor bends a roll angle. Finally the thrust is splitted in a vertical component and in a component in the E_a^2 axis.

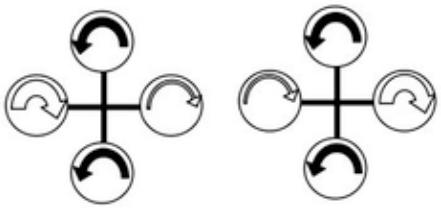

Figure 3.3: Left-Right movement

3. Move up-down: To move up or down it is necessary that the sum of all vertical thrust component for each rotor is more or less than the force due of the gravity.

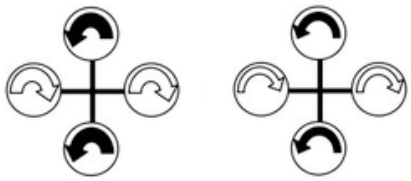

Figure 3.4: Up-Down movement

4. Rotate around E_a^3 (yaw motion): To rotate around the E_a^3 it should be a torque around this axis. There are two rotating clockwise and two counter-clockwise in order to compensate the torque created by the rotors. So the only way to change the yaw angles is unbalancing the torques generated by each rotor when they are spinning. As it is possible to check in the equation 3.6, the torque generated by the rotor it is controlled by its angular velocity. So changing the velocity of the rotors it is possible to reach to a determined yaw angle.

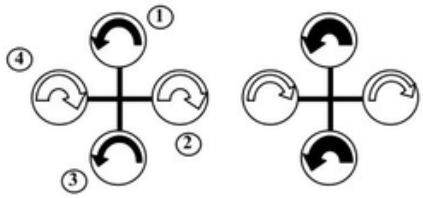

Figure 3.5: Rotate around E_a^3

By combining the fourth movement, it is possible to controls its position and its yaw angle. Due the dynamics of the system, in order to fix a position, it is necessary to minimize the pitch and roll angles.

As it is detailed above, the pitch and roll angles generate a deviation of the thrust and finally become in a motion in E_a^1 and E_a^2 .

On the other hand, the blade flapping effect also could change the vertical thrust and make new components in the E_a^1 and E_a^2.
The front side of the rotor disk is called the advancing side, and the left back side is called the retreating side. Blade flapping occurs when the rotors translate horizontally. In this case, a different lift between the advancing and retreating blades appears causing the rotor tip path plane tilts. In order to modelize this effect it should solve the constant and sinusoidal components of the blade centrifugal aerodynamic weight and find the tilt angle of the plane.

The flapping of the rotor it is found by calculating the magnitude and direction of the rotor's translation. It is necessary to define a new local reference frame located in the rotor and aligned in the direction of the rotor's movement (B_i).

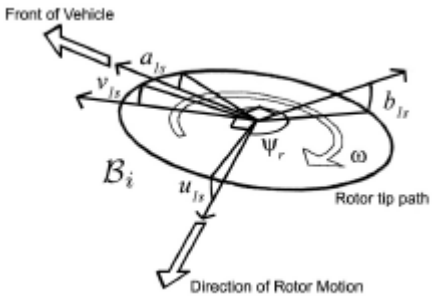

Figure 3.6: Parameters of a blade

In this frame, the longitudinal and lateral flapping angles are calculated (u_1s_i, v_1s_i) and then re-expressed them in the body frame of the quadrotor ($a1s_i$, $b1s_i$)
The expressions are the following:

$$v_{ri} = V + \Omega \times D_i \tag{3.8}$$

$$\mu_{ri} = \frac{\|v_{ri(1,2)}\|}{\omega_i \cdot R} \tag{3.9}$$

$$\Psi_{ri} = arctan(\frac{v_{ri(2)}}{v_{ri(1)}}) \tag{3.10}$$

3.8 Denotes the velocity vector of the i-th rotor. It is the sum of the velocity of the quadrotor plus the velocity due the angular rotation of the system.
3.9 It is the ratio between the magnitude of the advance velocity and the linear velocity of the blades. It is called advance ratio.
3.10 Is the azimuthal direction of the motion. On the other hand, the longitudinal and lateral flapping angles in the local reference B_i are:

$$u_{1s_i} = \frac{1}{1 - \frac{\mu_{ri}^2}{2}} \cdot \mu_{ri}^2 \cdot (4\theta_i - 2\lambda_i) \tag{3.11}$$

$$v_{1s_i} = \frac{1}{1 + \frac{\mu_{ri}^2}{2}} \cdot \frac{4}{3} \cdot (\frac{C_t}{\sigma} \cdot \frac{2}{3} \cdot \frac{\mu_{ri} \cdot \gamma}{a} + \mu_{ri}) \tag{3.12}$$

$$\lambda_i = \sqrt{\frac{C_T}{2}} \tag{3.13}$$

$$\gamma = \frac{\rho a_0 c r^4}{I_b} \tag{3.14}$$

where:
lade lift slope gradient otor radius

- σ: Rotor solidity is the ratio of the total blade area to the total disk area.
- t: Blade tip angle.
- a : Blade lift slope gradient.
- c : Blade chord.
- r : Rotor radius.
- I_b : Rotor blade rotational inertia about the flapping hinge.

3.11 is the longitudinal flapping angle. Depends on parameters of the blade and the inflow of the rotor.

3.12 is the lateral flapping angle. Depends on the geometry of the blade, the inflow of the rotor and the Lock Number.POUNDS et al. [2006]

3.13 Approximation of the Inflow of the rotor.

3.14 The Lock Number represents the ratio of the aerodynamic and inertial forces on the blade.

Most of these parameters are related with the geometry of the blades.

Once the longitudinal and lateral flapping angles are calculated in the local body frame of the rotor, they are transformed back to the quadrotor body frame using the frame mapping $J_{B_i}^A$

$$A_{m,n} = FALTAMATRIZ \tag{3.15}$$

Finally, the influence of the pitch and roll rates into the flapping blade effect are added to the flapping angles.

$$ALGO = J_{B_i}^A \cdot ALGO + ALGO \tag{3.16}$$

These values represents the effect of the geometry and physics properties of the propellers. Depending of the model and type of those blades, the behavior of the system varies.

Also, by modeling this effect, it is possible to correct some non controlled motion like the lateral movement when the quadrotor is hovering.

Still, the equations here presented are just a mathematical approximation of the physics of a blade where some assumption have been done JOHNSON [1994].

Using the model described above, it is possible to control the motion of the system by controlling the angular rate of each rotor and taking into account the geometry of the blades.

3.1.1.2 Control Model

To control the system through a NMPC it is necessary to have an accurate model of the system. The last model showed has been discretized and transformed into an algorithm to use it to predict the behavior of the quadrotor.

The equations 3.1 3.2 3.4 give us the derivative of the position and velocity, so in order to predict the position is necessary first discretize and then integrate for the sampling time:

1. Position:

$$\dot{\xi}(k) = \frac{\xi(k+1) - \xi(k)}{T_s} \rightarrow \xi(k+1) = \xi(k) + \dot{\xi}(k) \cdot T_s \tag{3.17}$$

$$\xi(k+1) = (x, y, z)^T(k+1) = (x, y, z)^T(k) + R \cdot V(k) \cdot T_s \tag{3.18}$$

2. Linear Velocity:

$$\dot{V}(k) = \frac{V(k+1) - V(k)}{T_s} \rightarrow V(k+1) = V(k) + \dot{V}(k) \cdot T_s \tag{3.19}$$

$$V(k+1) = (V_x, V_y, V_z)_B^T(k+1) = (V_x, V_y, V_z)_B^T(k) + \dot{V}(k) \cdot T_s \tag{3.20}$$

3. Euler Angles:

$$\dot{n}(k) = \frac{n(k+1) - n(k)}{T_s} \rightarrow n(k+1) = n(k) + \dot{n}(k) \cdot T_s \tag{3.21}$$

$$n(k+1) = (\psi, \theta, \varphi)^T(k+1) = (\psi, \theta, \varphi)^T(k) + \Omega(k) \cdot T_s \tag{3.22}$$

4. Angular Velocity:

$$W^{-1} \cdot \dot{\Omega} = \frac{\Omega(k+1) - \Omega(k)}{T_s} \rightarrow \Omega(k+1) = \Omega(k) + W^{-1} \cdot \dot{\Omega}(k) \cdot T_s \tag{3.23}$$

$$\Omega(k+1) = (\Omega_x, \Omega_y, \Omega_z)_I^T(k+1) = (\Omega_x, \Omega_y, \Omega_z)_I^T(k) + W^{-1} \cdot \dot{\Omega}(k) \cdot T_s \tag{3.24}$$

w^{-1}: Is the inverse of the wronskian. This matrix transforms the euler angles

rates to angles rates of the inertial frame.

On the other hand, the rates of each rotor are also discretized by taking a sample and holding it during all the sampling time.

These equations allows us to predict the position of the quadrotor for each sample time. This is a part of the system, in the next chapters, the effect of the robotic arm will be added to this system.

3.1.1.3 Simulation Model

Once the NMPC optimizes the problem, the control variables are returned in order to be applied to the real system.

The framework of this study is limited to simulation scenarios so it is necessary to have a model to emulate the real behavior and to obtain the states variables for the next horizon of control. Usually, in a real environment, this values are calculated using an estimator with sensor fusion algorithms.

On the other hand, it is important that the model used for the controller is different from the model used to simulate the quadrotor.In this way it is possible to tune the NMPC to absorb the possible errors and increase its robustness.

So the model to simulate is a simplification of the real model. In this case, all the flapping effects will be removed it from the dynamics equations. $a1s_i = 0$ $b1s_i = 0$

$$\dot{\xi} = R \cdot V \tag{3.25}$$

$$m \cdot \dot{V} = -m \cdot \Omega \times V + m \cdot g \cdot R^T \cdot E_3^a + \sum_1^4 T_i \tag{3.26}$$

$$\dot{\Omega} = -\Omega \cdot sk\left(\Omega\right) \tag{3.27}$$

$$I \cdot \dot{\Omega} = -\Omega \times I \cdot \Omega + \sum_1^4 \left[Q_i + M_i\right] \tag{3.28}$$

$$T_i = C_T \cdot \rho \cdot A \cdot r^2 \cdot \omega_i^2 \cdot [0, 0, -1)]^T \tag{3.29}$$

$$Q_i = C_Q \cdot \rho \cdot A \cdot r^3 \cdot \omega_i \cdot |\omega_i| \cdot E_3^a \tag{3.30}$$

$$M_i = T_i \times D_i \tag{3.31}$$

Finally, in order to use it, it should be discretized and integrated like the section above.

In the NMPC section has been written a full description about how the models has been used.

3.1.2 Real quadrotor

To obtain a realistic behavior of the quadrotor is important to adjust the physics and geomatics parameters to real values. Focused on this aim, most of the parameters have been taken from the quadrotor used by the researcher group that has motivated this study.

This team has been working with a modification of the Pelican UAV from Ascending Technologies

Most of the parameters that are used during this thesis come from other studies BRESCIANI, GRUENE and PANNEK [2011]

Some of them are experimental and were out of the scope of this research, so they couldn't be verified. On the other hand, these kind of parameters are taken from other studies that have checked its authenticity. Also, some of them are compared with the values given by the specification of the Pelican in its datasheet.

This parameters are splitted in several categories:

1. Physics: Gravity, viscosity and density of the air are defined to compute all the physical equations.

2. Airframe: Its mass and its inertial matrix have been defined in this section. Also the horizontal and vertical displacement of the rotors relative to the CoG.

3. Rotor: All the parameters about the blade geometry and its physics are described in this category. Here is where all the flapping blade effect are determined in function of the type of blade used by the quadrotor. Also its mass and its inertial matrix are detailed.

4. Constants: Some constants are precalculated to optimize the code during the control loop.

By changing this parameters and using the dynamics equations showed in this thesis, it is possible to simulate the behavior of any kind of X-type quadrotor.

The versatility of this method has allowed us to create an algorithm that is independent of the quadrotor used during the control process.

The parameters chosen are a mix of different models of quadrotors from different research groups, so do not correspond to any of them.

3.2 Robotic arm description

In this section the the parameters and the physics that define the behavior of a robotic arm will be detailed.

3.2.1 Mathematical model of the robotic arm

The aim of this section is to explain how to this study has obtain the dynamics equations of the robotic arm. First, it is necessary to explain the parametrization of the arm and then how to calculate the dynamic reaction in its base.

Any robotic arm with a system of serial joints it is possible to be represented by the Denavit-Hartenberg parameters.

These variables (also called DH parameters) are four values that define a particular convention of how to attach the reference frame of the links for a chain of

joints.

This convention consists on attach a coordinate frame of a link to on of its joints. For each joint there is a matrix transformation that allows to change from one frame to the next or previous one. Concatenating these transformation matrix it is possible to relate the end-effector frame with the base frame.

Depending of the type of the joint (hinge or sliding), one of the four parameters is the variable value and the other are constants due that they are a geometric distance or angle.

The algorithm to select the right frames S_i that fulfill the DH parameters is:

1. The z-axis is in the direction of the joint axis. I.e., the rotation axis or the displacement axis.
2. The x-axis is the parallel to the common normal $x_n = z_{n-1} \times z_n$. The direction of x_n is from z_{n-1} to z_n
3. The y-axis is selected to get the frame coordinate that accomplish the right-handed rule.

The four parameters are :

d_i**:** Offset along previous z to the common normal. $Distance_{z_{i-1}}(S_{i-1}, z_{i-1} \cap x_i)$

θ_i**:** Angle about previous z from old x to new x $Angle_{z_{i-1}}(x_{i-1}, x_i)$

a_i**:** Length of the common normal. For a revolute joint, is the radius about previous z $Distance_{x_i}(z_{i-1} \cap x_i, S_i)$

α_i**:** Angle about common normal, from the old z to new z. $Angle_{x_i}(z_{i-1}, z_i)$

The DH algorithm determines the configuration of the robotic arm and gives to us an unambiguous description of the system. Also, knowing the angles of each joint it is possible to exactly calculate the position of the end-effector relative to its base.

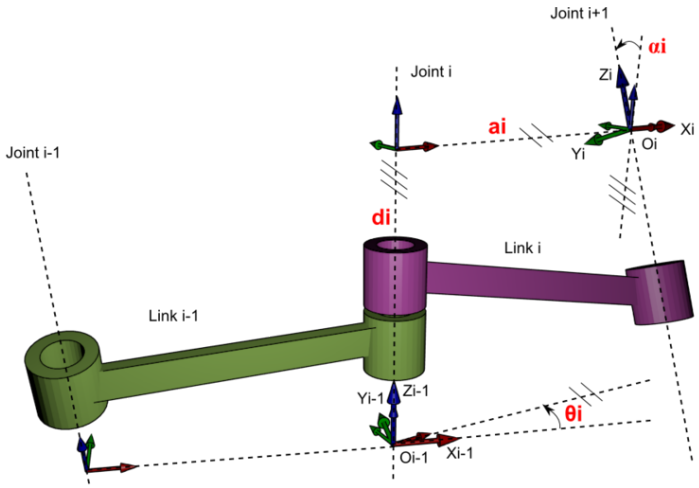

Figure 3.7: Denavit hartenberg parameters

This information will be important to estimate the dynamics reactions into the base due the motion of the end effector. The final target is to get these forces and torques to apply them to the quadrotor as a perturbation of the system.

The method selected to get the dynamics equations is the Recursive Newton-Euler (RNE) algorithm applied in a robotic arm. MELCHIORRI OLLERO [2011]

The RNE has two steps; forward process allows to calculate the linear and angular velocity of each link by going from the base to the end-effector. Backward process uses those velocities to solve the dynamics equations to obtain the force and torque of each joint from end-effector to base. So finally it is possible to get the reaction in the base due the motion of the system.

The formulation for the case of revolution joints are:

1. Forward step:

   - Angular Velocity. The angular velocity of the frame k is:

$$\omega_k = \omega_{k-1} + \dot{q}(k) \cdot z_{k-1} \tag{3.32}$$

     where ω_{k-1} is the vector of the angular velocity of the previous frame, $\dot{q}(k)$ is the joint velocity and z_{k-1} is the revolution axis.

- Angular Acceleration. The angular acceleration of the frame k is:

$$\dot{\omega}_k = \dot{\omega}_{k-1} + \dot{q}^2(k) \cdot z_{k-1} + \Omega_{k-1} \times (\dot{\omega}_k \cdot z_{k-1}) \tag{3.33}$$

  where $\dot{\omega}_k$ is the vector of the angular acceleration of the frame k and $\dot{q}^2(k)$ is the joint acceleration.

- Linear Velocity. The linear velocity of the frame k is:

$$v_k = v_{k-1} + (\omega_k \times \triangle s_k) \tag{3.34}$$

  where v_{k-1} is the linear velocity of the of the frame $k-1$. The $S_k = d_k - d_{k-1}$ is the vector difference between the position of the frame $k-1$ with the current frame k. Note that to calculate linear velocity it is necessary have the angular velocity of the frame, so the order is mandatory.

- Linear Acceleration. The linear acceleration of the frame k is:

$$\dot{v_k} = \dot{v_{k-1}} + (\dot{\omega_k} \times \triangle s_k) + \omega_k \times (\omega_k \times \triangle s_k) \tag{3.35}$$

  where $\omega_k \times (\omega_k \times \triangle s_k)$ is the centrifugal acceleration. Note that again to calculate linear acceleration is necessary to have the angular acceleration.

Finally, to start the iteration, initial conditions have to be set:

ω_0**:** Initial angular velocity of the base.

$\dot{\omega}_0$**:** Initial angular acceleration of the base.

v_0**:** Initial linear velocity of the base.

$\dot{v}_0$**:** Initial linear acceleration. Usually the gravity.

2. Backward step:

   - Forces:
     Let's define a vector from the end of a link to its CoG $\triangle r_k = \overline{c_k} - d_k$ where $\overline{c_k}$ is the location of the center of mass of link k.

The resulting force applied in the CoG is:

$$f_k = f_{k+1} + m_k \cdot [\dot{v}_k + \dot{\omega}_k \times \triangle r_k + \omega \times (\omega \times \triangle r_k)] \tag{3.36}$$

- Torques:
  First it should be calculated the moment of the frame k.

$$n_k = n_{k-1} + (\triangle s_k + \triangle r_k) \times f_k - \triangle r_k \times f_{k+1} + D_k \cdot \dot{\omega}_k + \omega_k \times (D_k \cdot \omega_k) \tag{3.37}$$

  Where D_k is the inertial tensor of link k in base space.
  Once the moment it is solved, the torque it is computed as:

$$\tau_k = n_k^T \cdot z_{k-1} + b_k \cdot \dot{q}_k \tag{3.38}$$

  Where z_{k-1} is the axis of the revolution and b_k is the viscous friction coefficient.

Once the backward step is finished the forces and torques reactions are obtained and the RNE is finished.

There are some geometrical and physical parameters that define the dynamic model of the system. Inertial tensors, viscosity or distance between frames are some examples.

In order to implement the RNE using the DH parameters a toolbox of CORKE [2011] has been used. This tool allows us to use some algorithms to calculate the RNE or the acceleration of each joint given the current angles of the joints, its velocity and the torque desired. Also, it is possible to calculate the torque necessary to compensate the gravity effect.

For the simulation model, a discretized method has been implemented. To calculate the dynamics reactions of the base for each sampled time, it is necessary to have the angles, the velocity and the acceleration of each joint.

The method developed in this thesis to discretize the model and obtain the

dynamics reactions is:

1. Calculate the coriolis velocity using the currents angles and velocities of the joints.
2. Estimate the necessary torque to be applied to each joint to compensate the effect of the gravity for the current position of the joints.
3. Compute the necessary torque to compensate the centrifugal force due the coriolis effect.
4. Solve the direct kinematics using the current angles, velocities and the torques desired for each joint. This torques contemplate the enough torque to compensate the gravity and coriolis effect, and also to achieve the motion planned. After solve it, the accelerations for each joints are obtained.
5. Update the joint velocity using the acceleration and the sampling time.
6. Update the joint angle using the velocity and the sampling time.
7. Finally, using the updated angles, velocities and accelerations, the Recursive Newton-Euler algorithm is applied. After this step, the base forces and torques reactions are obtained.

This algorithm is repeated for each sampled time. Due the complexity of the operations, it is an expensive method that spends a lot of CPU time.

The implementation of each step of this algorithm was done by using the Peter Corke Robotics Toolbox.

Once it is finished, the result is a vector with three forces and three moments (one per axis) represented in its base frame.

This information will be used during the control loop as a perturbation of the system. In the next chapters the relation between this model, quadrotor system and NMPC will be detailed.

3.2.2 Real Robotic Arm

This thesis has used the robotic arm developed by IRI on collaboration with european project Arcas SANTAMARIA and ANDRADE [2014]. Due this factor, it was possible to obtain all the Denavit-Hartenberg parameters of the model.

On the other hand, parameters like tensors matrix, coefficient of viscosity or the exact position of the CoG has been estimated.

As it has been said before, this robotic arm is a prototype and its final design will be different in comparison with the current model. However, the algorithm developed during this thesis are general and can be applied to any robotic arm. So, even if the model changes, only by changing the parameters of the model, the method explained in this study will work properly.

The robotic arm designed by IRI has 6 joints set in different ways in order to reach to any orientation. The reference frame follows the same convention as the body frame of the quadrotor:

1. $xaxis$: Aligned with the forward of the quadrotor.
2. $yaxis$: Pointing to the floor (same direction as the gravity).
3. $zaxis$: Completes the reference frame with the right-hand convention.

The 1st one is put on the base of the arm to rotate it along the $z - axis$ and move all the other articulations to any location of the xy plane . The next two joints move the rest of the joints in the plane xz. Finally, the last three joints are to orientate the end-effector independently of other joints.

During the simulation, the motion of each joint are controlled with a PID properly tuned. The end effector is empty, so all the dynamics are calculated with a no tool in the arm.

Because the method developed in this thesis is modular, it does not matter which robotic arm is used, but due the design is known it in the IRI it was possible to obtain most of the real parameters for the simulation.

Link i	a_i	d_i	α_i	θ_i
1	0.030	-0.026	$\pi/2$	q_1
2	0.079	0	0	$q_2 + \pi + 0.325$
3	0.0158	0	$\pi/2$	$q_3 + \pi - 0.325$
4	0	-0.118	π	q_4
5	0	0	$-\pi/2$	q_5
6	0	0	0	$q_6 + \pi/2$

Table 3.1: DH parameters of the robotic arm

3.3 Nonlinear Model Predictive Control problem

During this section, all the parts of the Nonlinear Model Predictive Control have been detailed

The aim of a NMPC is to calculate the optimal control signals to carry the plant to a given state or follow a trajectory of its state variables.

$$min_u J = \sum_{1}^{4} \gamma_i \cdot \frac{(x_i - x_{id})^2}{x_{iupperbound} - x_{ilowerbound}} \tag{3.39}$$

st.:

$$x(k+1) = f(x(k), u(k)) \tag{3.40}$$

$$x_{lowerBound} < x(k+1) < x_{upperBound} \tag{3.41}$$

$$u_{lowerBound} < u(k+1) < u_{upperBound} \tag{3.42}$$

$$-max \triangle u < x(k+1) < max \triangle u \tag{3.43}$$

$$x(0) = x_0 \tag{3.44}$$

$$u(0) = u_0 \tag{3.45}$$

The algorithm has several parts that works together in order to accomplish its function.

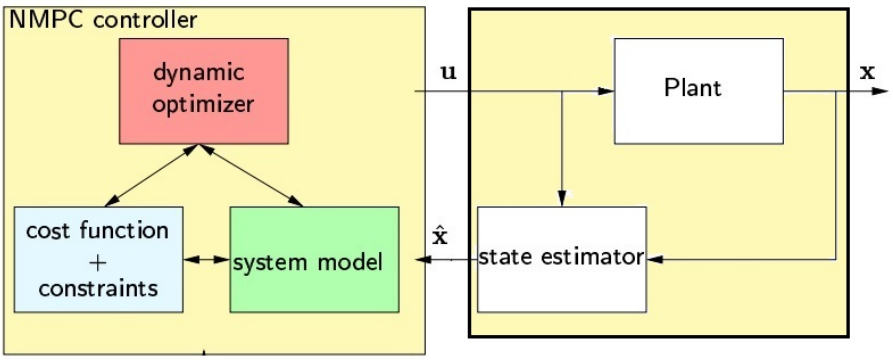

Figure 3.8: NMPC scheme

Plant: System to be controlled.

State Estimator: Make a estimation of the state variables.

Dynamic Optimazer: Solve the optimal problem.

System Model: Mathematical model of the behavior of the plant.

Cost Function: Function that should be minimized.

Constraints: Restrictions that should be respected to solve the optimal problem.

3.3.1 Dynamic model, state variables and control signals

The dynamic model of the whole system includes the quadrotor and the effect of the motion of the robotic arm.

This thesis develops a method to couple the effect of the arm with the quadrotor dynamics. First of all is important to specify the state variables chosen for this study.

The aim of this project is to control location and orientation of the quadrotor so the variable states are :

$(x_1, x_2, x_3) = (x, y, z)$: Position relative to the inertial frame.

$(x_4, x_5, x_6) = (\psi, \theta, \varphi)$: Euler angles (yaw, pitch , roll).

$(x_7, x_8, x_9) = (\dot{x_1}, \dot{x_2}, \dot{x_3}) = (\dot{x}, \dot{y}, \dot{z})$: Linear velocity relative to the inertial frame.

$(x_{10}, x_{11}, x_{12}) = (\Omega_x, \Omega_y, \Omega_z)$: Angular velocity of the body frame.

The control variables are the signals able to control the whole plant. In this case, the controllable values are the angular rate of the rotors

$(u_1, u_2, u_3, u_4) = (\omega_1, \omega_2, \omega_3, \omega_4)$: Position relative to the inertial frame.

The sign of these variables determine if the propellers are rotating clockwise or counter-clockwise. This sign is important to compute the torque generated by its rotation.

On the other hand, the arm it is considered as a perturbation of the system, so the NMPC has not the task of control its trajectory. To add this perturbation it is necessary to calculate the RNE for each sampled time before to apply the NMPC.

In order to simulate the trajectory of the arm, it has been necessary to build a PID for each joint. These controller is enough to move a joint from an angle to another one in a smoothly way.

Once all the reactions are calculated, it is possible to simulate the behavior of the whole system.

To couple both dynamics, the discretized equations of the model of the quadrotor has to be modified:

$$m \cdot \dot{V} = -m \cdot \Omega \times V + m \cdot g \cdot R^T \cdot E_3^a + \sum_1^4 T_i - F_{ar} \tag{3.46}$$

$$I \cdot \dot{\Omega} = -\Omega \times I \cdot \Omega + \sum_1^4 [Q_i + M_i] - \tau_{ar} \tag{3.47}$$

with:

F_{ar}: Reaction forces of the arm.

τ_{ar}: Reaction torques of the arm.

Note the sign of these reactions. The negative sign is because the Peter Corke Toolbox gave us the reaction of the virtual anchor of the arm base, but the for the dynamics, it is necessary the opposite.

As it is described above, adding just a new parameter, it is possible to couple the perturbation of the arm in the dynamic behavior of the quadrotor. For each sampled time, the value of the Far and ar may change in function of the precalculated arm motion.

Once the reactions are calculated and the equations are modified, it is possible to start the NMPC loop to optimize the control of the system.

3.3.2 Horizon and Sampling Time

Choose a properly control horizon and sampling time is important to control the plant.

The horizon of the control is the number of steps that the NMPC evolves the system toward the future to find the optimal control taking into account the future predictions. As the horizon increases, the complexity to solve the problem augments, and the CPU time required increases dramatically. So a constraint to limit the number of the steps is the power of CPU.
On the other hand, a not enough control horizon, the dynamic of the system can not be predicted and then the control could be impossible.

The sampling time is the amount of time that the time advances for each step of the control horizon. If is too large, the dynamics of the system will occur faster than the NMPC could control. On the other hand, if the sampling time is too short, the future predicted is near of the current time and the inertias of the motion are not possible to be predicted.

So it is important to balance both parameters in order to have a far horizon of control but with a short sampling time to detect the dynamics of the plant and also taking into account the power of the CPU required.

After some experimentation and evaluating several parameters of the horizon

control and sampling time, a good balance has been found.

Depending of the control mode, the parameters could vary. These variations are because the dynamics effects could affect more or less according of the motion of the quadrotor and then it is necessary a larger or shorter horizon control to stabilize the plant.

In the Chapter 3, more detailed explanation has been written.

3.3.3 Objective function

The objective function, also called value function, is the expression that the NMPC has to minimize by looking the best control signals possibles.

This function is the global target of the quadrotor and describes how far is from its goal. The goal could be dynamic or a trajectory of the state variables. On the other hand, this equation does not show the performance to accomplish the minimization.

During this thesis, two different functions has been developed depending if the quadrotor has to stand still or if it has to move to some position.

The value function has subgoals for each subtarget that it has to accomplish. Each sub function has its weight in order to prioritize which ones are more important others. But to use a weight system, first is important to normalize them. To do it, this project has used the range of the constraint for each variable involved in each subfunction:

$$J = \sum_{1}^{p} \gamma_i \cdot J_i \tag{3.48}$$

with:

J: Total cost function.

γ_i: Weight of the subfunction $i - th$

J_i: Subfunction $i-th$

In the particular case of this thesis, there are up to 4 subfunctions:

$$J = \sum_{1}^{4} \gamma_i \cdot \frac{(x_i - x_{id})^2}{(x_{iMax} - x_{iMin})} \tag{3.49}$$

x_id: Desired value of x_i.

x_iMax: Maximum possible value of the state variable x_i.

x_iMin: Minimum possible value of the state variable x_i.

The state variables involved in the value function are $(x_1, x_2, x_3, x_4) = (x, y, z, \psi)$ corresponding of quadrotor's position and yaw.

On the other hand, pitch and roll are not included because if the system is forced to have a determined value of these states, directly implies that the quadrotor will move along its x or y axis.

As it was explained before, a tilt on its x or y axis generates a change of the direction of the thrust vector, and a component of force could appear in the horizontal plane. So it is important to let free the pitch and roll in order to allow the proper movement of the quadrotor.

When the system moves along an horizontal axis, it is mandatory to increase the difference of the rotor speed between two opposite rotors. But doing this, the total moment of the system is not compensated and starts to rotate in the z axis.

So it is necessary to fix the desired yaw angle to ensure that the plant orientation is constant. If this restriction is not imposed, it is possible to move from one position to another but losing the orientation due the change of rotation of the rotor when the quadrotor is trying to move.

On the other hand, nor linear nor angular velocity have been included due that it is impossible to hover a quadrotor if the system is not allowed to change

its velocity conveniently.

The NMPC uses this equation in order to find the optima control signals to minimize the function value, which means accomplish all the goals.

3.3.4 Constraints

The constraints define the state space of the problem. It is a polyhedric space where the NMPC has to find the optimal solution.

As bigger is this space, larger it is the search. So it is important to limit each variable with an upper and lower value. Also, if there are any linear or nonlinear relation between variables it should be implemented in the problem.

The constraints help to find the optimal and also limit all possibles solutions. Also, those restrictions allow to define the performance desired by the system.

While the cost function to minimize express the goal to reach, the constraints tell to NMPC how to accomplish its target.

During this thesis, the constraints have been categorized in two types:

1. State Variables Constraints: All the constraints related with the state variables are defined in this category. The upper and lower values have to be defined for each of the 12 state variables of the plant.

    - Position Bounds: Defines the volume allowed to move the quadrotor.

    $$(x_{1Upper}, x_{2Upper}, x_{3Upper}) = (x_{upper}, y_{upper}, z_{upper})$$

    $$(x_{1Lower}, x_{2Lower}, x_{3Lower}) = (x_{lower}, y_{lower}, z_{lower})$$

    The trajectory of the quadrotor have to stay inside this volume in order to find a solution of the problem. Also, this volume has to be at least enough big to contain the possible error obtained during the hover maneuver.

- Euler Angles Bounds: Defines the maximum and minimum euler angles allowed.

$$(x_{4Upper}, x_{5Upper}, x_{6Upper}) = (\psi_{upper}, \theta_{upper}, \varphi_{upper})$$

$$(x_{4Lower}, x_{5Lower}, x_{6Lower}) = (\psi_{lower}, \theta_{lower}, \varphi_{lower})$$

  Taking into account that the $x_4 = \psi$ is the yaw angle, the range of this variable have to be enough wide to contain all the possible orientations in the xy plane necessary to accomplish the targets of the quadrotor.

  On the other hand, the range of pitch ($x_5 = 0$) and roll ($x_6 = 0$) angles have to be enough small to permit a slight tilt that allows to move the quadrotor but without losing the control of the plant. For large values of pitch or roll, the system can rotate completely until the thrust points to the same direction as the gravity, which would be critical for the plant.

  To avoid this, a small range has been chosen.

- Linear Velocity Bounds: Defines the maximum and minimum linear velocity allowed. These are referenced to the inertial frame.

$$(x_{7Upper}, x_{8Upper}, x_{9Upper}) = (\dot{x}_{upper}, \dot{y}_{upper}, \dot{z}_{upper})$$

$$(x_{7Lower}, x_{8Lower}, x_{9Lower}) = (\dot{x}_{lower}, \dot{y}_{lower}, \dot{z}_{lower})$$

  Those constraints allow to smooth the behaviour of the quadrotor. Due the limit of its speed, the effect of its dynamics are less aggressive than for high speeds.

- Angular Velocity Bounds: Defines the range of the angular velocity of the quadrotor.

$$(x_{10Upper}, x_{11Upper}, x_{12Upper}) = (\Omega_{xUpper}, \Omega yUpper, \Omega zUpper)$$

$$(x_{10Lower}, x_{11Lower}, x_{12Lower}) = (\Omega_{xLower}, \Omega_{yLower}, \Omega_{zLower})$$

As before, it is important to define a narrow range of values in order to do not lose the control of the motion and avoid the possible problems with the dynamics effects.

2. Control signal Constraints: The restrictions of the control signal usually corresponds to the physics capabilities of the actuators. In this thesis the control signals are the angular rate of the rotors.

   So depending of the kind of motor used, its limits and its behavior could change.

   The basics aspects that it should be limited are:

   - Angular Rates of the Rotors: Defines the range of the angular velocity of the each rotor.
     The upper bounds correspond the maximum possible angular rate that the motor is able to turn.

     These constraints take into account the sign of the value because the velocity could be positive or negative depending if it turns clockwise or counter-clockwise.

$$u_{1Upper}(k) = \omega_{1Upper} > 0$$

$$u_{2Upper}(k) = \omega_{2Upper} < 0$$

$$u_{3Upper}(k) = \omega_{3Upper} > 0$$

$$u_{4Upper}(k) = \omega_{4Upper} < 0$$

     On the other hand, the lower bounds are the minimum values experimentally found to avoid lose the control of the system. Under these limits the quadrotor may fall down. So actually is not the minimum

angular speed of the rotors.

$$u_{1Lower}(k) = \omega_{1Lower} > 0$$

$$u_{2Lower}(k) = \omega_{2Lower} < 0$$

$$u_{3Lower}(k) = \omega_{3Lower} > 0$$

$$u_{4Lower}(k) = \omega_{4Lower} < 0$$

So:

$$\omega_{1Lower} < u_1(k) < \omega_{1Upper}$$

$$\omega_{2Lower} < u_2(k) < \omega_{2Upper}$$

$$\omega_{3Lower} < u_3(k) < \omega_{3Upper}$$

$$\omega_{4Lower} < u_4(k) < \omega_{4Upper}$$

- Angular Accelerations of the Rotors: Defines the range of the angular acceleration for each rotor.
  It is important to define the maximum possible acceleration because this constraint limits how fast could change the angular speed of the rotors and then the dynamics of the quadrotor.
  For wide ranges the maneuvers are faster but also more aggressive so the system could become uncontrolled.

  On the other hand, for a narrow ranges, change the state variables of the plant could be too slow and then it may be difficult to accomplish the goals.

  The maximum width of the ranges are determined by the physical properties of the rotor.

  To implement this restriction it is necessary to know the last angular rate of the rotor and compute the current increment:

$$\triangle u_i(k) = u_i(k) - u_i(k-1) < max\triangle \rightarrow u_i(k) < max\ \triangle + u_i(k-1)$$

$$-\triangle u_i(k) = -u_i(k) + u_i(k-1) < max\triangle \rightarrow -u_i(k) < max\triangle - u_i(k-1)$$

for i $\in$ [1,4].

Once all those constraints are applied, the states space of the system is defined and the optimizer can find the solution in the control space.

Chapter 4

Simulation Results

During this section, the results obtained are going to be presented.

The problem to solve consists to use an NMPC to control the dynamics of a quadrotor that is perturbed by external forces and torques.

Those perturbations are represented by the movement of a robotic arm.

In order to proceed to the simulations, some assumptions have been done:

- Enough computational resources: Let's suppose that the code and the processor are enough to compute all this calculations in real time.
- Accurate estimator of the state variables: Let's assume that the are a good filter that return to us a good estimation of the 12 state variables. During these tests, this filter is emulated by using a mathematical model of the quadrotor.
- Point of application: The force/torque point of application is in the origin of the body frame.
- Control the angular rate of the rotors: Let's suppose that is possible to control the speed of the rotors and their dynamics are instant.
- There are no wind: Let's assume a controlled environment.

- The sampling time is constant: The clock of the computer is constant and the algorithm is solved without any delay.
- Robotic arm motion: The angular joint position are going to have small increments.
- Empty end-effector. No tools attached to the end-effector.

To test the algorithm developed during the thesis, five different scenarios have been prepared:

1. Taking off while carrying a robotic arm: The quadrotor take off and flies until reach to a specific altitude. The arm stays in a safe position.
2. The quadrotor in hover mode and the robotic arm in a safe position: The aim of this test is to try if the system is able to keep a fixed position while its center of mass has changed.
3. The quadrotor in hover mode and the robotic arm moving toward a target position: This scenario is a demonstration about how the NMPC compensate the dynamics of the arm due its motion.
4. The quadrotor moving toward location and the robotic arm in a safe position: The target is show how the quadrotor can move to a fixed position while is carrying an object that changes its center of mass.
5. The quadrotor moving toward a location and the robotic arm moving toward a target position: This environment has been prepared to test the MPC against the dynamics of the quadrotor perturbed by the robotic arm.

During the next sections, a comparison between those modes are going to be presented. In each part, some statistics have been compared by changing some of the parameters of the NMPC to show how they affect to the system. To obtain the perturbation is necessary to simulate a motion of the arm. For these tests, the same arm motion are going to be applied in order to understand better the different behavior of the whole system . The motion of the arm could be or stay in a safe position, or move to a target pose. Both cases has different reactions. This

plot shows the reaction when the arm robotic stays in a safe position. The forces and torques are constant during all the simulation. Actually, the only remarcable force that is acting to the system is the force in the z axis which corresponds to the force of the gravity. Also, there is a slight moment in the y axis.

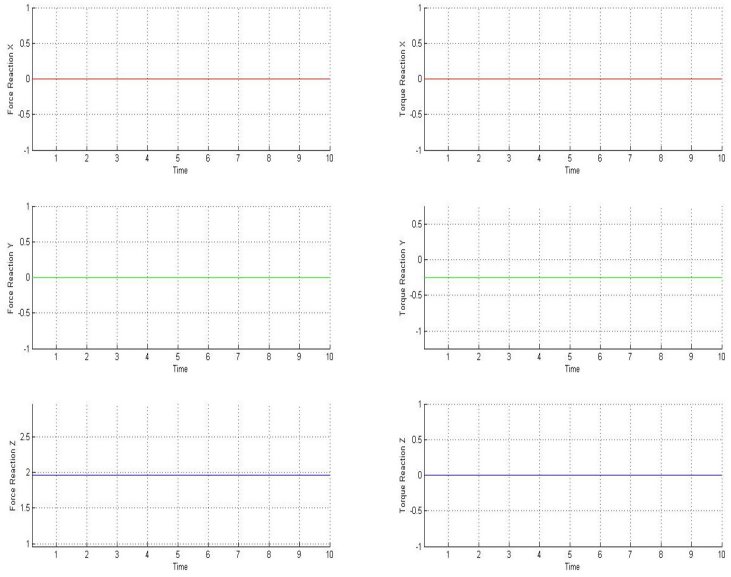

Figure 4.1: Dynamic Reaction in the base of the arm due its no motion

On the other hand, the motion chosen have been small step for each of the joints.

The graphic shows the forces that the quadrotor will receive when the robotic arm is moving. Those dynamics have to be compensate by the control algorithm in order to stabilize the system.

Depending on the scenario, the perturbations will change. It is possible to observe how when the angle reference change, appear forces and torques reactions until the position is reached. Once the arm it is stable, the dynamic reactions disappear.

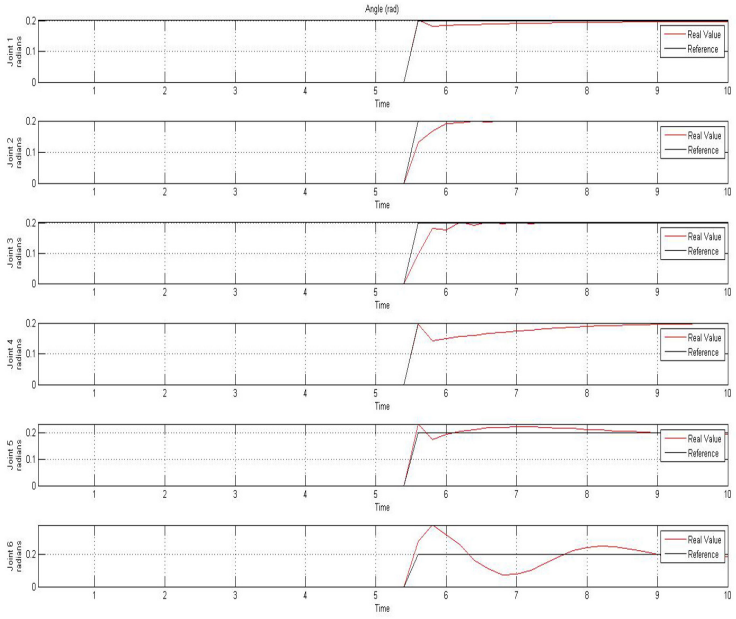

Figure 4.2: Reference and Angle of each joint

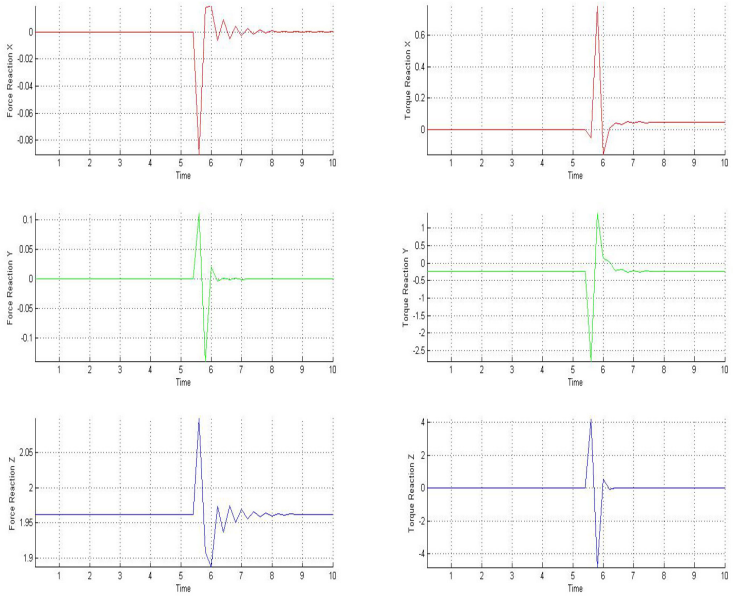

Figure 4.3: Dynamic Reaction in the base of the arm due its motion

4.1 Taking off

The test consists on take off the quadrotor until reach a safe position in the air. The robotic arm stays in a safe position during all this maneuver.

The whole system starts into the ground. The NMPC has to find the right combination of the inputs to take off.

In this case, the cost function is:
$if x_3 >= 0$

$$J = 10 \cdot \frac{(x_1 - 0)^2}{100} + 10 \cdot \frac{(x_2 - 0)^2}{100} + 10 \cdot \frac{(x_4 - 0)^2}{4} + 10 \cdot \frac{(x_9 + 1)^2}{100}$$

else

$$J = 10 \cdot \frac{(x_1 - 0)^2}{100} + 10 \cdot \frac{(x_2 - 0)^2}{100} + 2.5 \cdot \frac{(x_3 + 4)^2}{100} + 10 \cdot \frac{(x_4 - 0)^2}{4}$$

end with: $(x_1, x_2, x_3, x_4, x_9) = (x, y, z, \varphi, \dot{z})$ By using this strategy, the NMPC is forced to try push the quadrotor until obtain a velocity in the z axis.

Once the quadrotor starts to fly ($x_3 < 0$) will mean that has enough thrust to reach to the altitude desired, so the cost function changes to find the right location in the space.

Parameters	Value
Initial State Variables	$[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Desired State Variables	$[0, 0, -4, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Initial Control Signal	$[200, -200, 200, -200]$
Horizon Control	10
Sampling Time	$0.2s$
Simulation Time	$10s$

Table 4.1: Initial Conditions for Take Off

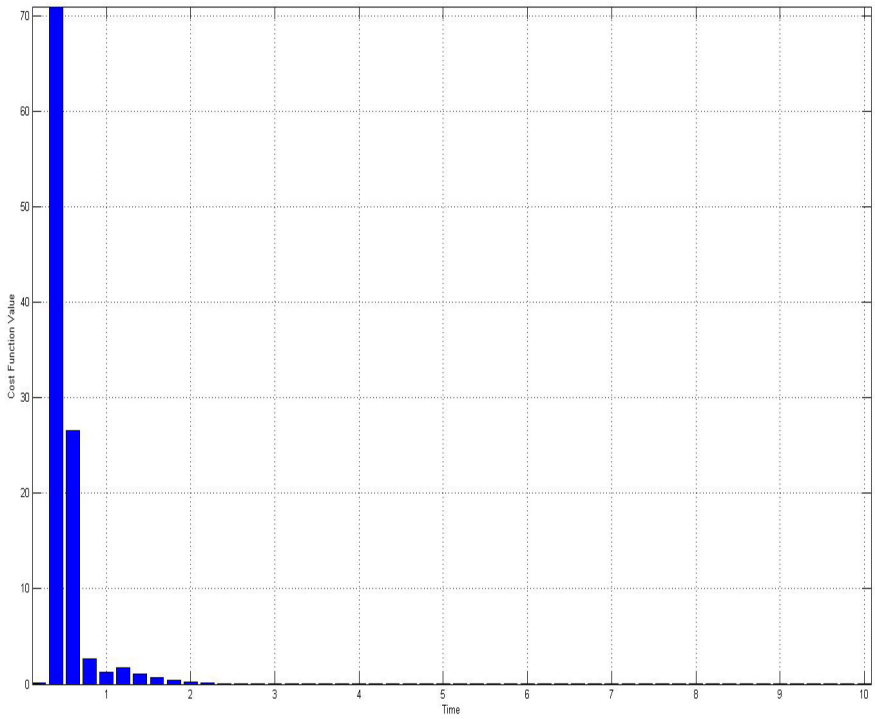

Figure 4.4: Cost function during the Take Off

As it is possible to see above, there is a peak. Before this points, is the step where the controller tries to push the quadrotor to move at $1m/s$ to up. After this point corresponds when the cost function has changed its branch, and the NMPC tried to reach to 4 m from the ground.

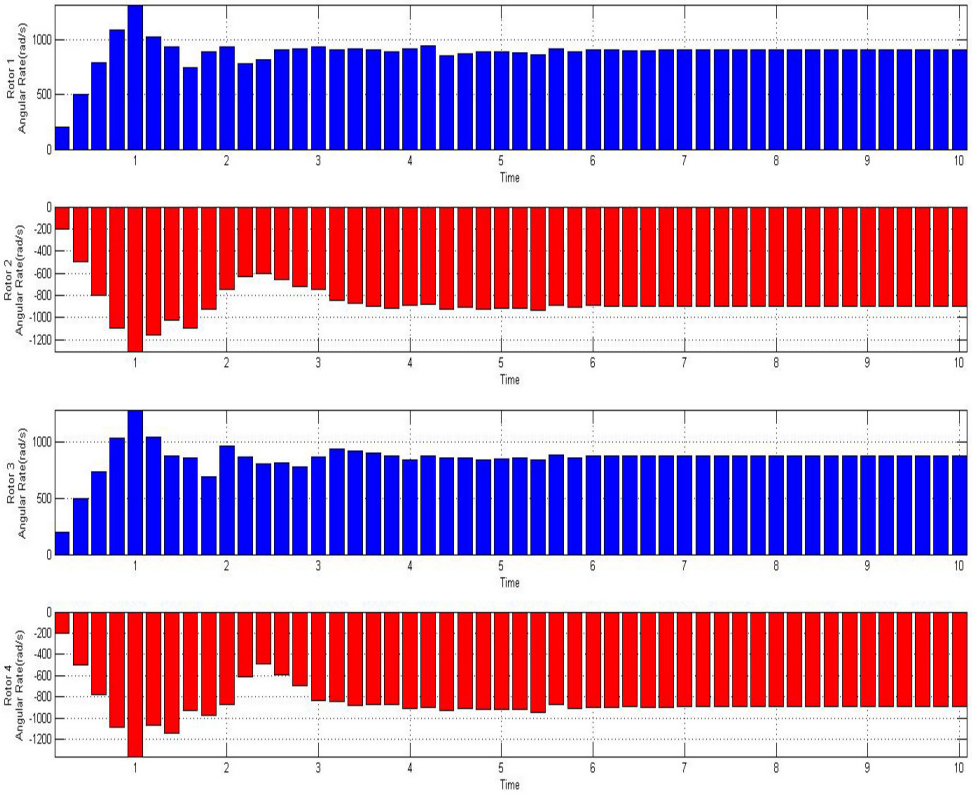

Figure 4.5: Control signal for a Take Off

At the beginning, the angular rate of the rotors increase its value in order to accelerate the quadrotor. Once it is flying, the control signal is looking for the way to stabilize the system. On the other hand, it is possible to notice the change of the behavior in the control signal when the quadrotor reach to the desired position.

Finally, the performance of the motion could be observed in the following graph.

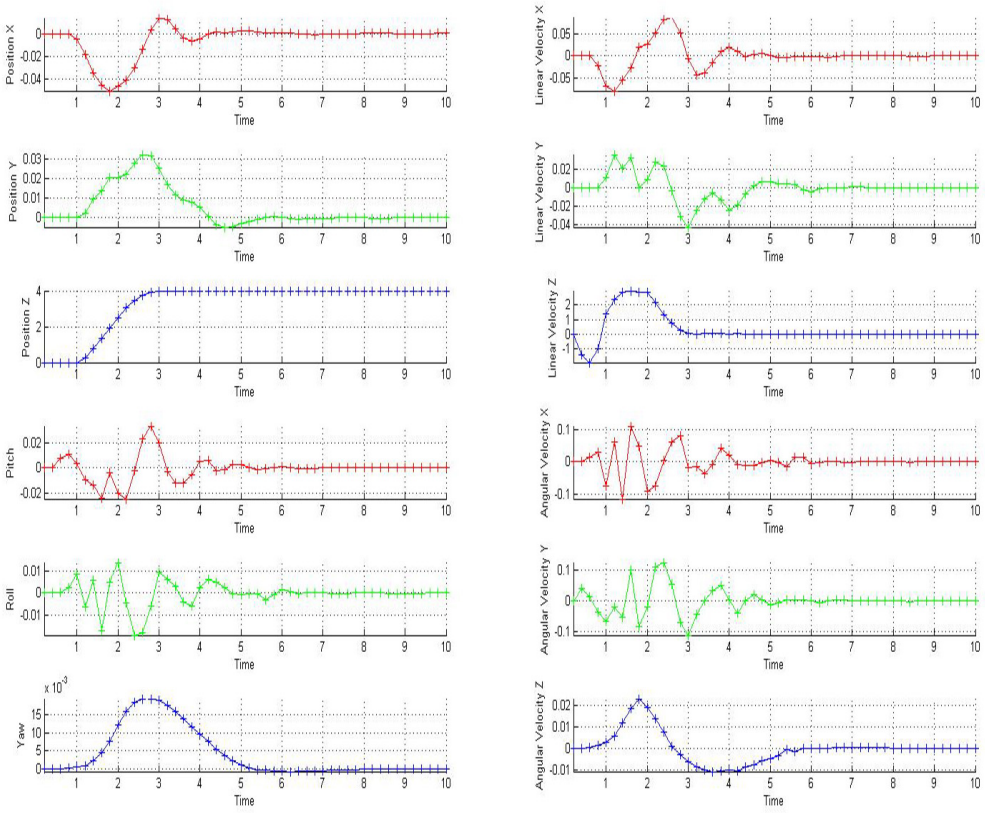

Figure 4.6: State variables for the Take off

The system, after reach to the enough velocity to take off, starts increasing its altitude until the $4m$. At the same time is compensating the movement in the xy plane and the tentative to change its yaw.

Parameters	Value
ISE	6.2
Maximum error	2.5
Time To Get Stable	$3.2s$

Table 4.2: Summary of Take Off

4.1.1 Static Control

The aim of this section is to try to stabilize the quadrotor in a fixed position in the space.

$$J = 10 \cdot \frac{(x_1 - 0)^2}{100} + 10 \cdot \frac{(x_2 - 0)^2}{100} + 10 \cdot \frac{(x_3 + 4)^2}{100} + 10 \cdot \frac{(x_4 - 0)^2}{4} \tag{4.1}$$

This cost function assure that the quadrotor will stay near the position $(0, 0, 4)$ and with a fixed orientation.

At the beginning, the rotors start with an initial angular speed $(900, -900, 900, -900)$. Those rates are the necessary to avoid the fall of the system.

Parameters	Value
Initial State Variables	$[0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Desired State Variables	$[0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Initial Control Signal	$[900, -900, 900, -900]$
Horizon Control	10
Sampling Time	$0.2s$
Simulation Time	$10s$

Table 4.3: Initial Conditions for Static Control

4.1.2 Static arm

This is the simplest test where the robotic arm stays in a safe position and the quadrotor has to compensate its perturbations.

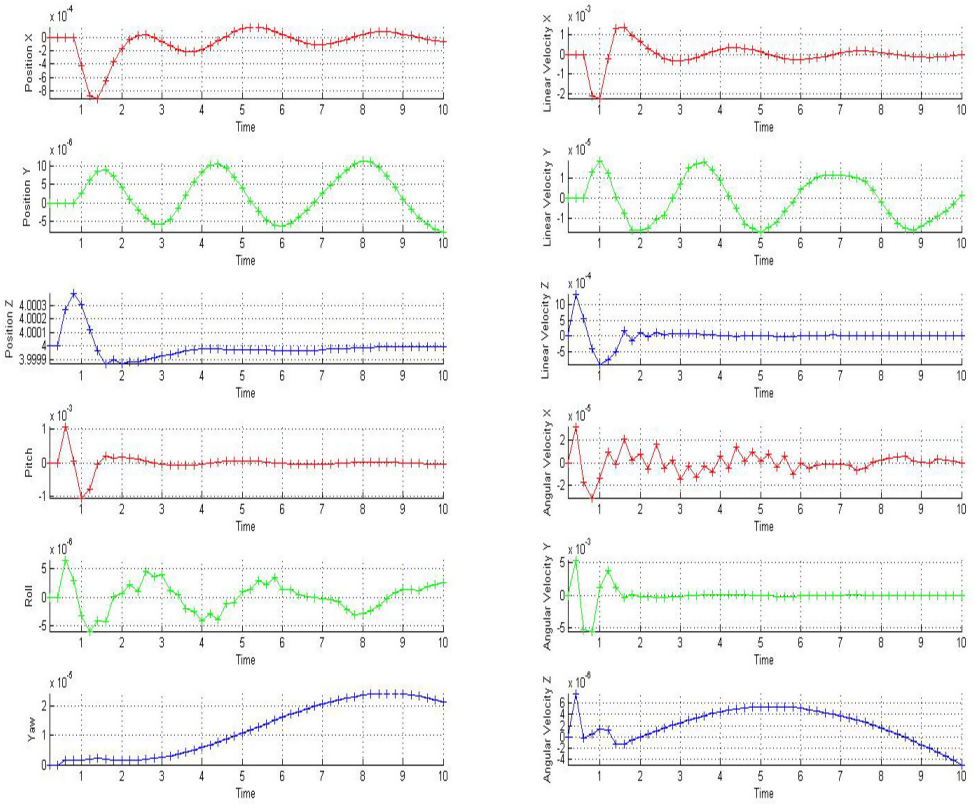

Figure 4.7: State variables a static control with a static arm

As it is possible to see, the NMPC could stabilize the quadrotor and bring it to the desired state variables.

In order to compensate the moment generated by the arm, the four rotors has different speed $(911.2652, -894.7493, 877.9722, -894.7513)$. Notice that the forward and backward rotors are unbalanced while the left and right are rotating at the same rate.

All the values are enough small to consider them as noise. It is clear to see

how the velocity in the y axis is trying to correct the deviation of the system in this axis.

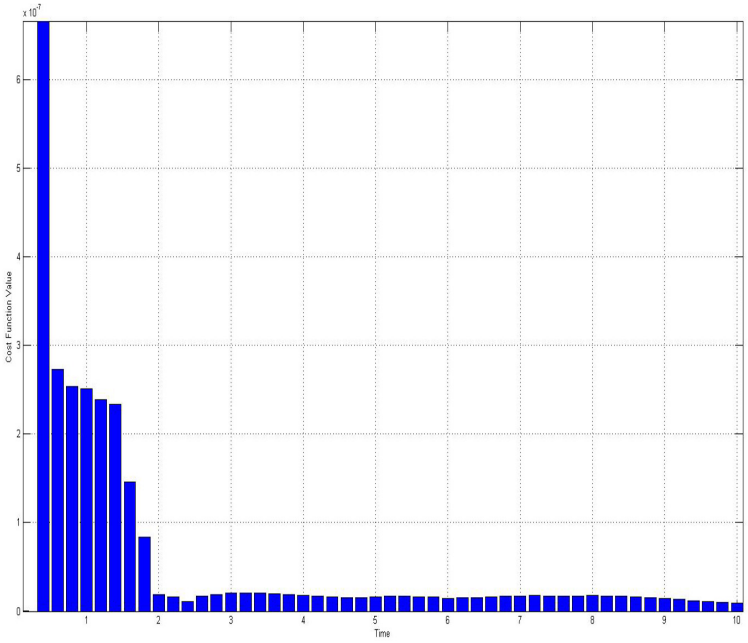

Figure 4.8: Cost function during the static control with a static arm

Approximately at 2 seconds, the system finds the correct combination of the rotor to reach at its stable position.

Parameters	Value
ISE	$2.75e^{-6}$
Maximum error	$6.75e^{-7}$
Time To Get Stable	$2.0s$

Table 4.4: Summary of a static control with a static arm

4.1.3 Dynamic arm

During this test, the perturbation of the arm robotic due its motion it has been applied to show its impact in the whole system.

4.2 Dynamic Control

The aim of this section is to try to carry the whole system to a position in the space.

$$J = 10 \cdot \frac{(x_1 - 0.75)^2}{100} + 10 \cdot \frac{(x_2 - 0.75)^2}{100} + 2.5 \cdot \frac{(x_3 + 4)^2}{100} + 10 \cdot \frac{(x_4 - 0)^2}{4} \quad (4.2)$$

The cost function has the target of move the quadrotor from the $(0, 0, 4)$ to $(1, 1, 4)$ and stay stable in this position.

Again the rotors start with an initial angular speed $(900, -900, 900, -900)$.

Parameters	Value
Initial State Variables	$[0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Desired State Variables	$[0.75, 0.75, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0]$
Initial Control Signal	$[900, -900, 900, -900]$
Horizon Control	10
Sampling Time	$0.2s$
Simulation Time	$10s$

Table 4.5: Initial Conditions for dynamic Control

4.2.1 Static arm

In this scenario, the quadrotors move to a position while the robotic arm stays immobile.

Parameters	Value
ISE	$2.75e^{-6}$
Maximum error	$7e^{-7}$
Time To Get Stable	$1.75s$

Table 4.6: Summary of a dynamic control with a static arm

4.2.2 Dynamic arm

Finally, the last case is to analyse how the perturbations from the robotic arm affects to the quadrotor while it is moving.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

5.2 Future Work

During the steps of this project new interesting topics have appeared to be studied in depth. Some of them are just improvement of the project here presented, and others are new fields of study to expand the control possibilities of the quadrotor.

In this way, a nice research could be optimize the code to implement it to a real system. In order to do it, would be important to reduce the computational time by applying a faster solver of the problem. Also, it would be interesting develop an algorithm able to calculate the minimum control horizon necessary to stabilize the current dynamics of the system. Updating this horizon, the system would always calculate only the necessary steps to control the quadrotor and a lot of operations would be avoided.

Other important task would be to study the effect of change the dynamic of the arm by modifying its parameters online.This effect would simulate when the arm starts to carry some object in its end-effector or if some joints start to fail.

Also it would be interesting to develop a method to change the application

point of the arm perturbations to a generic point of the quadrotor. In order to achieve it, it is necessary to study the effects due to apply a force and a torque in a different point of the CoG of the whole system.

Finally, another important research would be include the state variables of the joints and the end-effector inside the NMPC. By this technique would be obtained a more accurate control of the system. Also it would be possible to include a cost function of the end-effector to accomplish some specific tasks.

Here I put my conclusions ...

References

Tommaso BRESCIANI. Modelling, identification and control of a quadrotor helicopter. Master's thesis. 20

Peter. CORKE. *Robotics, vision and control: fundamental algorithms in MATLAB*. Springer, 2011. 25

Lars GRUENE and Juergen PANNEK. *Nonlinear Model Predictive Control. Theory and Algorithms*. Springer, 2011. 20

Wayne JOHNSON. *Helicopter Theory*. Dover, 1994. 17

Claudio. MELCHIORRI. Dynamic model of robot manipulators. Master's thesis. 23

Aníbal OLLERO. *Robótica. Manipuladores y robots móviles*. Marcombo, 2011. 23

Paul POUNDS, Robert MAHONY, and Peter CORKE. Modelling and control of a quad-rotor robot. *Proceedings Australasian Conference on Robotics and Automation*, 2006. 9, 17

Àngel SANTAMARIA and Juan ANDRADE. Hierarchical task control for aerial inspection. *euRathlon-ARCAS Workshop and Summer School on Field Robotics*, 2014. 27