Design and Implementation of Flight Dynamics Control Strategies for a Smartphone-based Quadrotor

Thesis for obtaining the degree of

Master of Science in Engineering
with emphasis in Automation

Alejandro Astudillo Vigoya
alejandro.astudillo@correounivalle.edu.co

School of Electrical and Electronic Engineering
UNIVERSIDAD DEL VALLE
Cali, COLOMBIA

November 20, 2017

Supervised by:

Dr.-Ing. Esteban Rosero
Industrial Control Research Group - GICI
School of Electrical and Electronic Engineering
Universidad del Valle

Bladimir Bacca Ph.D.
Perception and Intelligent Systems Research Group - PSI
School of Electrical and Electronic Engineering
Universidad del Valle

Abstract

The field of autonomous systems control is young, but operational experience is rapidly growing, making research on collaborative systems of great importance. Improving aerial robots in particular could be key in facing future environmental challenges..... In this work, two main problems are addressed: the cooperative source seeking problem and the cooperative level curve tracking problem by a group of agents under undirected constrained communications. .....

Resumen

The field of autonomous systems control is young, but operational experience is rapidly growing, making research on collaborative systems of great importance. Improving aerial robots in particular could be key in facing future environmental challenges..... In this work, two main problems are addressed: the cooperative source seeking problem and the cooperative level curve tracking problem by a group of agents under undirected constrained communications. .....

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Quadrotor control is a difficult and interesting problem. A quadrotor has six degrees of freedom (three translational and three rotational) and four independent inputs (forces applied by the motors). As established by Liu and Wang [2015] and Lopez et al. [2015], quadrotor dynamics are affected by nonlinearity, parameters perturbations, uncertainties and disturbances: this include unknown and variable payloads, aerodynamical parameters of the system, wind changes, and sensors inaccuracies. Numerous studies have been developed in designing optimal and robust controllers that allow unmanned aircraft systems (UAS) to fly and accomplish missions rejecting disturbances and being robust to parameter uncertainties as seen in Jung et al. [2014], Kohno and Uchiyama [2014], Shang et al. [2016], Salazar et al. [2014].

Although there are embedded systems with high computational capacity that can serve as controllers of a quadrotor, smartphones are available, easily accessible for people and also have a large computational capacity, hence multiple instrumentation and communication elements integrated in the same device. The use of smartphones also facilitates distribution and installation of updates of the control application as it is a commonly known device and has application distribution platforms. Some attempts to joint smartphones and aerial robots have been made: in Pearce et al. [2014], a smartphone was used as a mission planner for a quadrotor and in Bjälemark and Bergkvist [2014], a smartphone was used as flight controller using its sensors and power to stabilize the quadrotor and control its altitude. A smartphone has been used as a flight controller and processing system for image-based positioning in a quadrotor, as shown in Loianno et al. [2015a].

This research project aims to design and implement algorithms that will be executed in a smartphone to estimate and control quadrotor dynamics by the Research Group in Industrial Control. This project confronts several challenges such as using a smartphone as a hardware development platform, trying to use a non real-time operating system for real-time applications, designing optimal and high order con-

trollers using Java or C++, and executing that controllers in a smartphone.

This paper presents the design of an optimal and a robust controller, and a state estimator based on a Kalman filter for a smartphone-based quadrotor. The non-linear and linerized model of the quadrotor is presented, the controller that allows the quadrotor to follow a trajectory reference using two approachs is designed. The two approaches are the LQG and H_∞ controllers. The quadrotor flight dynamics estimation strategies using sensor fusion algorithms are described.

Current smartphone processors are able to perform complex calculations such as those required in the implementation of real time control strategies. There are many ongoing research related to the possibility of using smartphones to implement control strategies, such as Drumea [2013], as configuration and monitoring interfaces in control systems as seen in Lin et al. [2014], Truong and Vu [2012], and as a tool in both education and design of control strategies seen in Aristizabal et al. [2014], Wu Wu [2013]. Following this trend, in the Universidad del Valle, it was developed a smartphone-based platform for monitoring, control and communication in portable laboratories, where a controller for a pendulum based in the Lego Mindstorms EV3 platform was implemented García Téllez and Ochoa Calambás [2015].

In recent years, the interest in aerial robotics research has increased substantially. This is because this type of robotics offers several potential new services such as search and rescue, observation, mapping, inspection, etc. On the other hand, smartphones have become essential devices for humans and easily acquirable development tools. The interaction between these two technologies allow the development of low cost quadcopters based on an everyday item such as the smartphones, facilitating the distribution of the quadcopter control software and its implementation by other researchers.

In the University of Pennsylvania, in Loianno et al. [2015a], was developed a quadcopter using a last generation smartphone as a flight controller and an additional processing system for image-based positioning. The state estimation algorithms, control and planning were firstly implemented in a ODROID-XU board with additional sensors, but then, in Loianno et al. [2015b], this algorithms were ported to the Qualcomm processor in the phone due to the Qualcomm colaboration in that project.

Current research focuses on the development of aerial robots potentiated by the use of smartphones, as seen in Pearce et al. [2014], Bjälemark and Bergkvist [2014], Aldrovandi et al. [2015], Bryant et al. [2015]. In the last years, computing capacity and sensor technology in smartphones has decreased in price but increased in performance. Smartphones have become an inexpensive tool capable of commanding an UAV. The challenge then, is to use smartphones as quadcopter flight controllers

Figure 1.1: Assembled quadcopter used in this research with the on-board smartphone on the top center of it.

for autonomous flights following specific missions, taking advantage of the fact that the phones today are very powerful computers that include elements of sensing, processing and signal communication.

In this paper, the implementation of a quadcopter with a smartphone acting as its flight controller while using exclusively the sensors and processor in the smartphone, is shown. The controller keeps the attitude of the quadcopter stabilized while making the quadcopter to hover at an altitude reference. It is presented the detailed composition of the test platform (quadcopter) used, integrating it with its dynamic model in addition to the quadcopter altitude and attitude estimation strategies using sensor fusion algorithms.

1.1 Motivation

wfwfew

1.2 Research Problem

En los últimos años el interés por la investigación en robótica aérea ha aumentado sustancialmente. Esto se debe a que este tipo de robótica ofrece diferentes servicios potenciales como búsqueda y rescate, observación, mapeo, inspección, entre otros. Por otro lado, los teléfonos inteligentes se han convertido en dispositivos esenciales para el ser humano y en herramientas de desarrollo y prototipado rápido

de fácil acceso. La interacción entre estas dos tecnologías permitiría el desarrollo de cuadricópteros a bajo costo basados en un elemento cotidiano como los teléfonos inteligentes, facilitando la distribución del software de control y su implementación a manos de otros investigadores.

Los desafíos de investigación existentes están en cómo desarrollar e implementar algoritmos eficientes de control para teléfonos inteligentes usando el sistema operativo Android, y valorar, adaptar y desarrollar las tecnologías adecuadas de comunicación, sensado y actuación con los teléfonos inteligentes en la ejecución de misiones utilizando cuadricópteros.

La pregunta a responder es ¿cómo desarrollar estrategias de control en un teléfono inteligente para el control de vuelo de un cuadricóptero de manera que se utilice la instrumentación y capacidad de computación del teléfono y se puedan desarrollar misiones utilizando un cuadricóptero?

1.3 Objectives

Objetivo General

Para responder a la pregunta de investigación se plantea el siguiente objetivo general:
Diseñar e implementar algoritmos de control y estimación de dinámicas de vuelo ejecutados en un teléfono inteligente para el cuadricóptero del Grupo de Investigación en Control Industrial.

Objetivos Específicos

1. Realizar un estudio y análisis del estado del arte relacionado con el control y la estimación de los estados de cuadricópteros.

2. Integrar el cuadricóptero existente con un telefono intelingente que contenga los sensores adecuados para el control y la estimación de estados.

3. Obtener un modelo dinámico del cuadricóptero.

4. Diseñar e implementar los algoritmos de control y de estimación de estados para el cuadricóptero.

5. Integrar la plataforma de experimentación con los algoritmos de control y estimación en el teléfono inteligente.

6. Evaluar el desempeño de las estrategias de control.

1.4 Literature Review

wfwfewf

1.4.1 Quadrotors

wfwefwe

1.4.2 Quadrotor Flight Modes

http://ardupilot.org/copter/docs/flight-modes.html ArduPilot Dev Team [2016]

Stabilize Mode

Stabilize mode allows you to fly your vehicle manually, but self-levels the roll and pitch axis.

Pilot's roll and pitch input control the lean angle of the copter. When the pilot releases the roll and pitch sticks the vehicle automatically levels itself.

Pilot will need to regularly input roll and pitch commands to keep the vehicle in place as it is pushed around by the wind.

Pilot's yaw input controls the rate of change of the heading. When the pilot releases the yaw stick the vehicle will maintain it's current heading.

Pilot's throttle input controls the average motor speed meaning that constant adjustment of the throttle is required to maintain altitude. If the pilot puts the throttle completely down the motors will go to their minimum rate (MOTSPIN-ARMED) and if the vehicle is flying it will lose attitude control and tumble.

The throttle sent to the motors is automatically adjusted based on the tilt angle of the vehicle (i.e. increased as the vehicle tilts over more) to reduce the compensation the pilot must do as the vehicle's attitude changes.

Altitude Hold Mode

In altitude hold mode, Copter maintains a consistent altitude while allowing roll, pitch, and yaw to be controlled normally. This page contains important information

about using and tuning alt hold.

When altitude hold mode (aka AltHold) is selected, the throttle is automatically controlled to maintain the current altitude. Roll, Pitch and yaw operate the same as in Stabilize mode meaning that the pilot directly controls the roll and pitch lean angles and the heading.

Automatic altitude hold is a feature of many other flight modes (Loiter, Sport, etc) so the information here pertains to those modes as well.

The pilot can control the climb or descent rate of the vehicle with the throttle stick.

If the throttle stick is in the middle (40% 60%) the vehicle will maintain the current altitude. Outside of the mid-throttle deadzone (i.e. below 40% or above 60%) the vehicle will descend or climb depending upon the deflection of the stick. When the stick is completely down the copter will descend at 2.5m/s and if at the very top it will climb by 2.5m/s.

Loiter Mode

Loiter Mode automatically attempts to maintain the current location, heading and altitude. The pilot may fly the copter in Loiter mode as if it were in a more manual flight mode but when the sticks are released, the vehicle will slow to a stop and hold position.

A good GPS lock, low magnetic interference on the compass and low vibrations are all important in achieving good loiter performance.

The pilot can control the copter's position with the control sticks.

Horizontal location can be adjusted with the the Roll and Pitch control sticks with the default maximum horizontal speed being 5m/s (see Tuning section below on how to adjust this). When the pilot releases the sticks the copter will slow to a stop. Altitude can be controlled with the Throttle control stick just as in AltHold mode The heading can be set with the Yaw control stick

Return-To-Launch Mode

When RTL mode is selected, the copter will return to the home location. The copter will first rise to RTLALT before returning home or maintain the current altitude if the current altitude is higher than RTLALT. The default value for RTLALT is 15m.

RTL is a GPS-dependent move, so it is essential that GPS lock is acquired before attempting to use this mode. Before arming, ensure that the APM's blue LED is solid and not blinking. For a GPS without compass, the LED will be solid blue when GPS lock is acquired. For the GPS+Compass module, the LED will be blinking blue when GPS is locked.

RTL will command the copter to return to the home position, meaning that it will return to the location where it was armed. Therefore, the home position is always supposed to be your copter's actual GPS takeoff location, unobstructed and away from people. For Copter if you get GPS lock and then ARM your copter, the home position is the location the copter was in when it was armed. This means if you execute an RTL in Copter, it will return to the location where it was armed.

Auto Mode

In Auto mode the copter will follow a pre-programmed mission script stored in the autopilot which is made up of navigation commands (i.e. waypoints) and "do" commands (i.e. commands that do not affect the location of the copter including triggering a camera shutter). This page provides an overview of Auto mode.

1.4.3 Smartphones as Controllers

wfwfwewf

1.4.4 Smartphone-based Quadrotors

wfwfwf

Smartphone-based Quadrotor Limitations

The idea of using a smartphone as flight controller in an UAV opens the possibility of a rapid and inexpensive development Aldrovandi et al. [2015]. A smartphone offers other advantages compared with off-the-shelf flight controllers, for instance its powerful quad or hexa-core processors and communications interfaces. However, smartphones and the Android operating system have some limitations that set challenges when implementing a control system in it.

Android is not a real-time operating system and thus can not assure execution of algorithms, like estimation and control, with a constant sample time. Furthermore, the sensors embedded in commercial smartphones are made for applications

that do not have high requirements of accuracy nor precision and therefore may not be appropriate for sensing quadrotor dynamics. Nonetheless, as explained by Bryant et al. [2015], due to its computing capabilities, smartphones can overcome this limitations while using a temporized thread to execute the control system algorithms and implement a sensor fusion technique to improve the states estimation reliability. This thread must be executed with a lower sample time compared to the one of the sensors embedded in the smartphone. This will ensure that the execution is not delayed by the sensors acquisition process.

1.5 Outline

In the next section, a detailed description of the hardware used in the quadcopter is presented. After that, the dynamic non-linear and linearized quadcopter model is shown in Section **??**. In Section **??**, all the aspects related to the altitude and attitude cascade controllers design are presented. Then, in Section **??** the sensor fusion algorithm that is used to estimate the altitude of the quadcopter, and how the orientation of the system is estimated, is described. In Section **??**, the implementation in Android of all the necessary software with its results is provided. Finally, Section **??** concludes this paper taking into account the results of this research and the future work to be done.

This paper is organized as follows: The smartphone-based quadrotors features and limitations are shown in Section **??**. In Section **??**, the dynamic model of a quadrotor in X-configuration using a Newton-Euler approach is presented. After that, in Section **??**, the quadrotor parameters and controllers design procedure is exposed. In Section **??**, the state estimation algorithms based on a Kalman filter and the kinematic model of a particle with constant acceleration are provided. The performance of the proposed flight control system through simulations is presented in Section **??**. Finally, Section **??** shows the conclusions and future aiming of this project.

Chapter 2

Dynamic Model of the Quadrotor

In this chapter, the derivation of the quadrotor model is provided. This result is very important because it describes how the helicopter moves according to its inputs. Thanks to these equations it is possible to define and predict the positions reached by the helicopter by investigating just the four motor speeds. The model equations will be "inverted" in the next chapter (Control algorithms) to identify which inputs are needed to reach a certain position.

The first section (2.1: Quadrotors' Configurations) shows the main idea of the quadrotor dynamics and describes intuitively which movements are allowed and how it manages to perform stationary flight (hovering).

The second section (2.2: Quadrotors' Inputs) provides the model information with physics and mathematical derivations. In this work, the Newton-Euler formalism and the Euler angles theories have been chosen.

In the third section (2.3: Nonlinear Model), additional information was added to the model taking into account the whole motor system which is composed of the motor itself, the reduction gears and the propeller.

The last section (2.4: Linearized Model) provides an overview of the architecture: connections between devices and abstraction of the software and task.

2.1 Quadrotors' Configurations

As the term 'quadrotor' refers to a multirotor whose thrust is generated from four motors and propellers, quadrotors can be built in multiple ways as long as they comply with the established definition. There are some configurations that are already standardized being widely used by commercial manufacturers and hobbyists,

such as the '+' and 'X' configurations.

2.1.1 '+' Configuration

The geometry used in quadrotors built in '+' configuration is shown in Fig. 2.1.

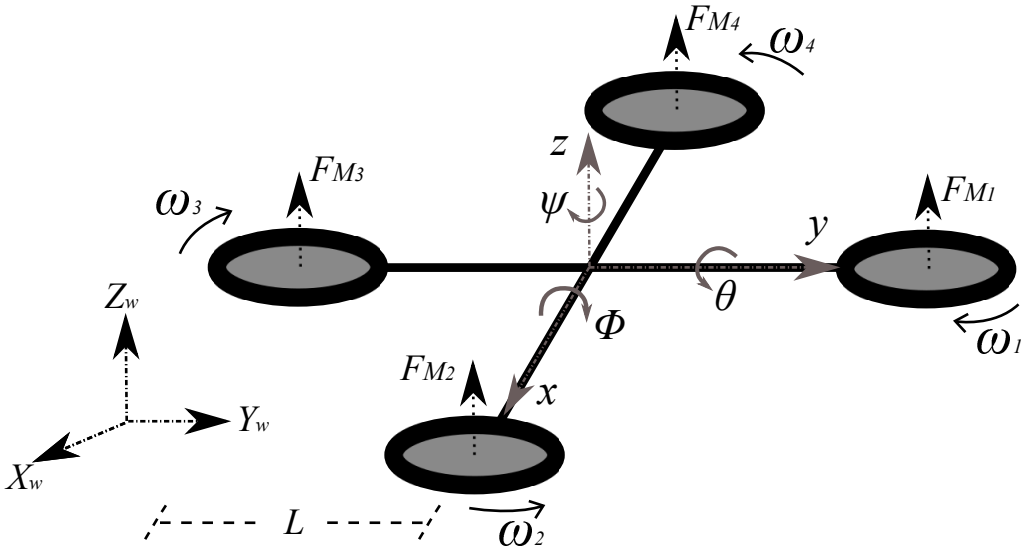


Figure 2.1: Quadrotor geometry in '+' configuration

Where, (ϕ, θ, ψ) are the angular deviations (pitch, roll and yaw, respectively) of the quadrotor about the body-frame, (x, y, z) are the body-frame axes, (X_W, Y_W, Z_W) are the earth-frame axes, L is the distance from the quadrotor's center of gravity (CoG) to the motor center, F_{M_i} is the thrust force exerted by each motor, and ω_i is the angular velocity of each motor, with $i = 1, 2, 3, 4$.

In this configuration, the body-frame axes x and y, coincide with the lines that connect motors of opposite sides in the quadrotor's frame. This means that, in order to change the pitch or roll angles, only the angular velocities ω_i of two motors must be modified, (ω_1, ω_3) for pitch, and (ω_2, ω_4) for roll. Here, M_1 is considered as the front motor, M_3 as the rear one, M_2 as the right motor, and M_4 as the left motor.

As can be seen in Fig. 2.1, M_1 and M_3 have a clockwise rotation while M_2 and M_4 rotate counter-clockwise. This configuration of opposite pairs rotational directions allows the system to control its conservation of momentum and thus change its yaw angle in a controlled manner without the need of a tail rotor used in the standard helicopter structure (Bresciani [2008]). In '+' configuration, the fact that

the pitch and roll angles are controlled using only two motors that rotate in the same direction, leads to large changes in the thrust force of the other two motors to achieve momentum conservation.

2.1.2 'X' Configuration

In 'X' configuration, the quadrotor's frame is rotated $\pi/4$ *rad* about the z-axis in the body-frame with respect to the '+' configuration, as shown in Fig. 2.2.

In this case, the front-line in the quadrotor is set between M_1 and M_4.

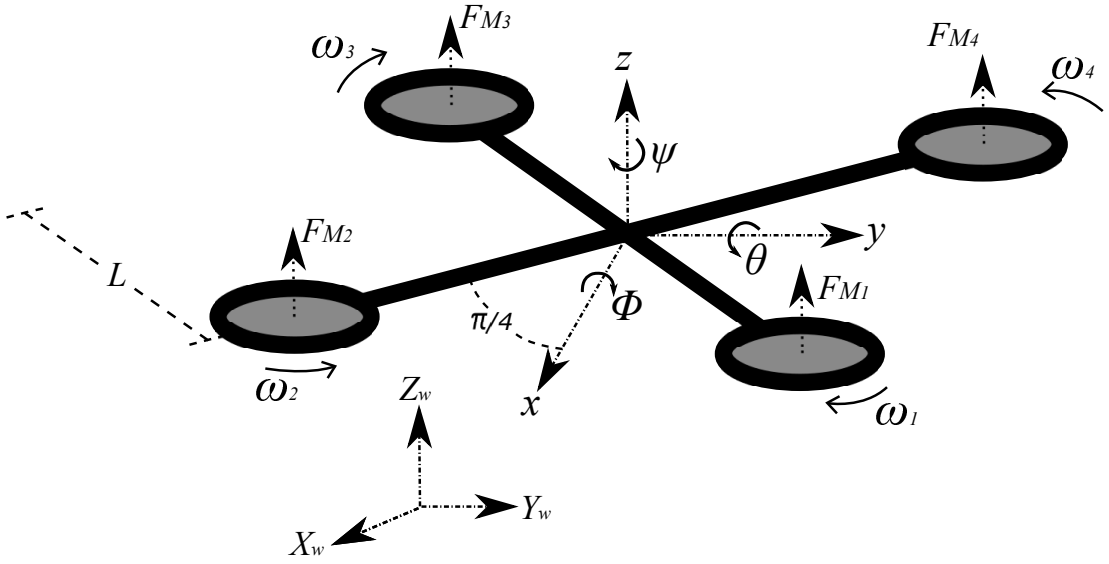


Figure 2.2: Quadrotor geometry in 'X' configuration

2.2 Quadrotors' Inputs

In this section,

2.2.1 Inputs Description

iyhhkhjjkhkjhhjk

Throttle (u $[N]$)

Yaw Torque (τ_ψ $[N \cdot m]$)

Roll Torque (τ_θ $[N \cdot m]$)

Pitch Torque (τ_ϕ $[N \cdot m]$)

2.2.2 Inputs Setting in '+' Configuration

$$\tau = \begin{bmatrix} \tau_\psi \\ \tau_\theta \\ \tau_\phi \end{bmatrix} = \begin{bmatrix} K_m(F_{M_2} + F_{M_4} - F_{M_1} - F_{M_3}) \\ L(F_4 - F_2) \\ L(F_3 - F_1) \end{bmatrix}, \tag{2.1}$$

2.2.3 Inputs Setting in 'X' Configuration

$L_X = L \cdot \cos(\pi/4)$ is the real distance between the point of application of the rolling and pitching torques and the quadrotor's center of mass along the x and y axes Faessler et al. [2016].

The rolling, pitching and yawing torques contained in vector τ, are generated using the force exerted by each motor as

$$\tau = \begin{bmatrix} \tau_\psi \\ \tau_\theta \\ \tau_\phi \end{bmatrix} = \begin{bmatrix} K_m(F_{M_2} + F_{M_4} - F_{M_1} - F_{M_3}) \\ L_X(F_3 + F_4 - F_2 - F_1) \\ L_X(F_2 + F_3 - F_1 - F_4) \end{bmatrix}, \tag{2.2}$$

where T_i is the torque produced by each motor along the z_b axis, L is the distance between each motor's rotor and the quadrotor's center of mass, and L_X is the real distance between the point of application of the rolling and pitching torques and the quadrotor's center of mass along the x_b and y_b axes Faessler et al. [2016].

2.3 Nonlinear Model

This section describes the dynamic modeling used to perform the quadrotor control, based on the study carried out in **???**. This model represents the quadrotor as a solid symmetrical object subject to a total thrust and three torques, without considering the dynamics of the actuators.

2.3.1 Euler-Lagrange Approach

The general coordinates representing the position and attitude of the quadrotor are defined as

$$q = \begin{bmatrix} \xi & \eta \end{bmatrix}^T, \tag{2.3}$$

where $\xi = \begin{bmatrix} x & y & z \end{bmatrix}^T$ is the vector representing the position of the center of mass of the quadrotor relative to the body reference frame shown in Fig. **??** and $\eta = \begin{bmatrix} \psi & \theta & \phi \end{bmatrix}^T$ represent the quadrotor's attitude.

The Lagrangian of the quadrotor is defined by

$$L(q, \dot{q}) = K_{trans} + K_{rot} - U, \tag{2.4}$$

where $K_{trans} = \frac{m}{2}\dot{\xi}^T\dot{\xi}$ is the translational kinetic energy, $K_{rot} = \frac{1}{2}\dot{\eta}^T J\dot{\eta}$ is the rotational kinetic energy, $U = mgz$ is the potential energy, m is the quadrotor's mass , z is the quadrotor's elevation, g is the gravity acceleration magnitude, and J is the inertial matrix. The dynamic model of the quadrotor is derived from the Euler-Lagrange equation

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \begin{bmatrix} F_\xi \\ \tau \end{bmatrix}, \tag{2.5}$$

where $F_\xi = R_b^w \hat{F}_b$ is the translational force applied to the quadrotor by the four motors, τ contains the rolling, pitching and yawing torques, and

$$R_b^w = \begin{bmatrix} C_\theta C_\psi & C_\psi S_\theta S_\phi - C_\phi S_\psi & S_\phi S_\psi + C_\phi C_\psi S_\theta \\ C_\theta S_\psi & S_\psi S_\theta S_\phi + C_\phi C_\psi & C_\phi S_\psi S_\theta - S_\phi C_\psi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix} \tag{2.6}$$

is the rotation matrix from the body to the Earth frame where $C_\theta = \cos\theta$ and $S_\theta = \sin\theta$.

In the quadrotor's body frame, the translational force $\hat{F}_b$ is only applied in the z_b axis as shown in Fig. **??**. This force is represented by

$$\hat{F}_b = \begin{pmatrix} 0 \\ 0 \\ u \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} F_{M_i} \end{pmatrix}, \tag{2.7}$$

with F_{M_i} being the force, in N, exerted by the motor M_i, as shown in Fig. **??**.

The force F_{M_i} has a linear dependency with the square of the motor angular velocity, defined as

$$F_{M_i} = k_i w_i^2, \tag{2.8}$$

where w_i is the angular velocity of the motor, and k_i is a proportional constant. However, in practice F_{M_i} must be set using the PWM signal input of an ESC. The thrust-PWM relation is found experimentally and is shown in Section **??**.

The Euler-Lagrange equations can be divided in two parts, one for the ξ coordinates and another for the η coordinates, getting

$$\ddot{\xi} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \frac{u_1}{m}(C_\phi S_\theta C_\psi + S_\phi S_\psi) \\ \frac{u_1}{m}(C_\phi S_\theta S_\psi - S_\phi C_\psi) \\ \frac{u_1}{m}(C_\phi C_\theta) - g \end{bmatrix}, \tag{2.9}$$

$$\ddot{\eta} = \begin{bmatrix} \ddot{\psi} \\ \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} \dot{\phi}\dot{\theta}\frac{J_{xx} - J_{yy}}{J_{zz}} + \frac{u_2}{J_{zz}} \\ \dot{\phi}\dot{\psi}\frac{J_{zz} - J_{xx}}{J_{yy}} + \frac{u_3}{J_{yy}} \\ \dot{\theta}\dot{\psi}\frac{J_{yy} - J_{zz}}{J_{xx}} + \frac{u_4}{J_{xx}} \end{bmatrix}, \tag{2.10}$$

where, $\begin{bmatrix} u_1, & u_2, & u_3, & u_4 \end{bmatrix}^T = \begin{bmatrix} u, & \tau_\psi, & \tau_\theta, & \tau_\phi \end{bmatrix}^T$, and (J_{xx}, J_{yy}, J_{zz}) are the moments of inertia around the quadrotor's body axes Emam and Fakharian [2016], Badr et al. [2016].

that is a simplified representation of the quadrotor complete model found in Bouabdallah [2007].

2.3.2 Newton-Euler Approach

sdasdasdaasd

2.4 Linearized Model

2.4.1 Jacobian Linearization

Sabatino [2015]

The Euler-Lagrange equations in (2.9) and (2.10) are linearized using their Jacobian

around the hover state where $\left[\eta,\ \dot{\eta},\ \dot{\xi}\right] \rightarrow \left[0,\ 0,\ 0\right]$, getting

$$\ddot{q} = \begin{bmatrix} g\theta \\ g\phi \\ u_1/m \\ u_2/J_{zz} \\ u_3/J_{yy} \\ u_4/J_{xx} \end{bmatrix}, \tag{2.11}$$

As said above, in order to perform the linearization, an equilibrium point is needed. Such an equilibrium point can be:

$$\overline{\mathbf{x}} = \begin{bmatrix} \overline{x} & \overline{y} & \overline{z} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{2.12}$$

From the equations, we can find that the equilibrium point (2.29) is obtained by the constant input value:

$$\overline{\mathbf{u}} = \begin{bmatrix} mg & 0 & 0 & 0 \end{bmatrix}^T \tag{2.13}$$

with mg being the lift force.

The linearized model of the quad-rotor helicopter written as a state space model is given by

$$\begin{aligned} \dot{\mathbf{x}}(t) =& A\mathbf{x}(t) + B\mathbf{u}(t), \\ \mathbf{y}(t) =& C\mathbf{x}(t), \end{aligned}$$

where

$$A = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}\Big|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.14}$$

$$B = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}\Big|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{zz}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{yy}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{xx}} \end{bmatrix}^T,$$

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

with the parameters

$m = 0.64$ kg,

$g = 9.81$ m/s.

The state vector is defined as

$$x(t) = \begin{bmatrix} r_x & \dot{r}_x & r_y & \dot{r}_y & r & \dot{r}_z \end{bmatrix}^T,$$

and the control inputs as

$$u(t) = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T,$$

and the output vector is defined as

$$r(t) = \begin{bmatrix} r_x & r_y & r_z \end{bmatrix}^T.$$

2.4.2 Thrust Compensation

https://robotics.stackexchange.com/questions/4247/tilt-compensated-motor-output-tc
Recalling the rotation matrix R_b^w,

$$R_b^w = \begin{bmatrix} c\theta c\psi & c\psi s\theta s\phi - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta \\ c\theta s\psi & s\psi s\theta s\phi + c\phi c\psi & c\phi s\psi s\theta - s\phi c\psi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2.15}$$

$$u = u^* \cos\theta \cos\phi \tag{2.16}$$

$$u^* = \frac{u}{\cos\theta \cos\phi} \tag{2.17}$$

2.5 Conclusions

This chapter presented the design of the two vehicles developed in this thesis. The test-bench and the OS4. The first system is only capable of 3 DoF which facilitates the testing of the controllers. However, it is possible to detach the flying part in order to test free flights. Before designing the second system which is a free flying quadrotor, a new design methodology is introduced. It allows an optimal design of small-scale rotorcraft. Four new design indicators were introduced for a precise and complete evaluation of the design performance. This methodology appreciably facilitated the components selection process and battery dimensioning of OS4. This quadrotor exhibits higher capabilities and endurance than the competition. This is verified through the comparison of different design parameters. OS4 embeds all the necessary avionics and energy devices for a fully autonomous flight. This comprises a low cost IMU, a vision based position sensor specifically developed for this project and an obstacle detection setup.

Chapter 3

Smartphone-based Quadrotor Prototype

In this chapter, the quadrotor prototype is presented. All the components that are used to build the prototype are described with the purpose of detailing the function that each component fulfils within the quadrotor. Furthermore, the specific parameters of the built quadrotor are shown and the procedure carried out to find them experimentally is explained. This is of great importance for the design of the controllers that allow the flight of the built system.

In Fig. 3.1, a small overview of the hardware related to the electrical signals within the quadrotor is presented. This overview shows the interrelation between the main components detailed below.

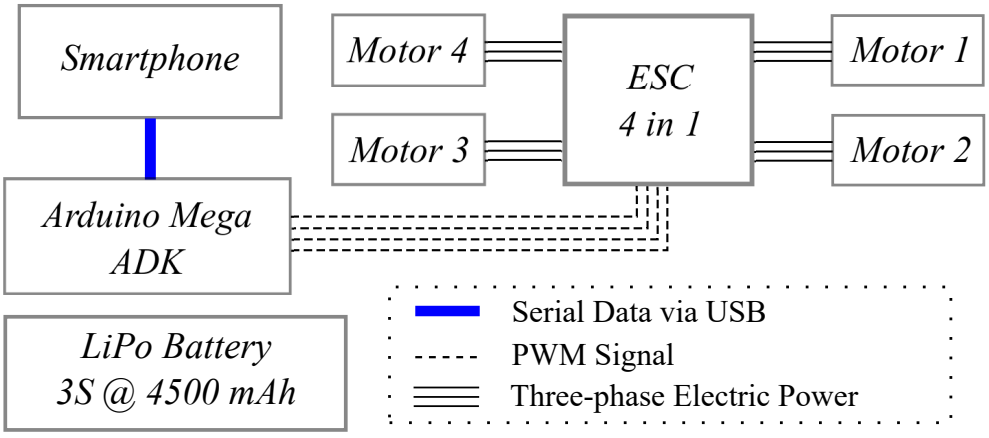


Figure 3.1: Quadrotor prototype's hardware overview

3.1 Quadrotor Components

The hardware used to build the quadrotor prototype is detailed in this section. The main goal of this research project is to control a quadrotor using just a smartphone on-board. This smartphone executes the state estimation algorithms and defines the controls signals that will command the actuators. The control signals are processed in the smartphone and then sent, through a gateway, to the Electronic Speed Controllers (ESCs) that will set the angular speed of the motors depending on the control signal they receive.

3.1.1 Frame

The quadrotor's frame is responsible for carrying all the other components needed in the system. Additionally, it must be able to support the weight of any possible payload that the quadrotor will carry. The multirotors frames are usually built using fiberglass or carbon fiber as main material, being the carbon fiber frames stiffer and lighter than equally built fiberglass frames.

For this project, the requirement to build a quadrotor between 0.20 and 0.30 cm in radius was defined. This requirement is based on the fact that the quadrotor must carry a smartphone and possibly some mission-related payloads, which weight can be greater than 100 g. Taking that requirement and the literature review into account, in addition to the weight and stiffness advantages of the carbon fiber, a LJI 500-X4 carbon fiber frame was selected.

Figure 3.2: LJI 500-X4 carbon fiber frame[1]

This frame (Fig. 3.2) has a weight of 431 g and a radius L of 0.244 m measured between the frame center and each of the four rotor axis holes. Its landing gear

[1]LJI 500-X4 frame image taken from https://goo.gl/hHfHQR

has a height of 0.170 m, allowing the payload to be placed in the lower part of the quadrotor.

3.1.2 Smartphone

In this project, the smartphone takes the place of the quadrotor's flight controller. The basic instrumentation needed for a flight controller includes a triaxial accelerometer, a triaxial gyroscope, a triaxial magnetometer, a barometer and a GNSS receiver that can process signals from the GPS and GLONASS satellites constellations. On the other side, the processor of a flight controller must be powerful enough to execute the control and estimation algorithms within the sample time of the control system.

The LG Nexus 5X, shown in Fig. 3.3, is a smartphone developed by Google and assembled by LG, released in 2015 with the Android 6.0.1 operating system. This phone has a Hexa-core Qualcomm MSM8992 Snapdragon 808 CPU that includes four Cortex-A53 and two Cortex-A57 cores with 1.4 GHz and 1.8 GHz clock rate respectively and an Adreno 418 GPU. Its features also include 2 GB of RAM memory, 32 GB of Flash memory, total mass of 136 g and a 12.3 MP camera.

Figure 3.3: LG Nexus 5X, smartphone used as flight controller[2]

This smartphone features all the needed instrumentation for a flight controller, specified previously. The maximum sample rates of the instrumentation contained in the LG Nexus 5X are described in Table 3.1.

[2]LG Nexus 5X image taken from https://goo.gl/RxyDJd

Table 3.1: Sample Rates of the Sensors in the Smartphone

Sensor	Sample rate $[Hz]$
Triaxis accelerometer	400
Triaxis gyroscope	200
Triaxis magnetometer	50
Barometer	10
GNSS	1

The LG Nexus 5X smartphone was selected in this project mainly due to its computational and instrumentation capabilities in addition to its communication interfaces (Bluetooth 4.0, Wireless LAN, USB Type-C and GSM). Its powerful CPU handling a GPU enables the possibility of developments using the camera for visual odometry, and controllers whose execution represents a high computational load, which exceeds the limits of a standard microcontroller such as the ATmega 2560.

As the quadrotor's flight controller, the smartphone receives remote orders from the quadrotor's pilot using wireless communication, while executing the sensorial, estimation and control algorithms. After the control signals are calculated, they are sent to the actuators as a serial data frame using the USB interface between the smartphone and a gateway that converts the serial data into PWM signals.

3.1.3 Motors and Electronic Speed Controllers

The actuation system of the quadrotor is composed by R/C-type brushless motors, propellers and ESCs. The LJI 500-X4 frame manufacturer recommends using motors with a RPM constant of 810 KV, which means that these motors can rotate at a speed of 810 RPM for each Volt applied to it. This kind of motors are used in medium to big sized multirotors due to its thrust efficiency and thrust capacity.

Following this recommendation, the EMAX MT2216II motors were selected. These motors, shown in Fig. 3.4a, are labelled by its manufacturer as 810 KV motors with a maximum current consumption of 9.8 A and a maximum thrust of 6.6 N, when using a 10-inch propeller.

(a) EMAX MT2216II motor

(b) EMAX 4in1 ESC

Figure 3.4: Motors and ESC used in the Quadrotor[3]

The quadrotor's brushless motors are powered using a three-phase electric signal generated by the ESC. Each ESC sets one motor rotational velocity depending on a PWM signal input, and must be able to handle the maximum current consumption of the motor. Mainly due to the ease of handling and positioning within the frame, the EMAX 4in1 ESC was selected for this project (Fig. 3.4b). This 4in1 ESC contains four ESCs with a maximum current supply of 30 A each. It also has a DC-DC converter that outputs an isolated 5 V DC supply for any additional electronics. The power supply of the ESC is fully delivered by the quadrotor's battery.

3.1.4 Smartphone-to-ESC Gateway

As the smartphone can not generate any PWM signal and each ESC needs a PWM signal input to set the rotational velocity of a motor, a gateway between the smartphone and the ESCs must be used. In order to avoid interference in the electromagnetic spectrum and delays that could lead the control system to instability while using wireless communications, the smartphone wired serial bus (USB interface) is defined as the only channel of communication through which the control signals are sent to the gateway.

The Arduino Mega ADK, is shown in Fig. 3.5, is a development board based on the Atmel 8-bit AVR RISC-based ATmega2560 microcontroller. This board supports the Android Open Accesory (AOA) protocol, which allows external hardware to exchange data with Android devices.

[3]Taken from https://goo.gl/6qD6Qg and https://goo.gl/fDHiUp

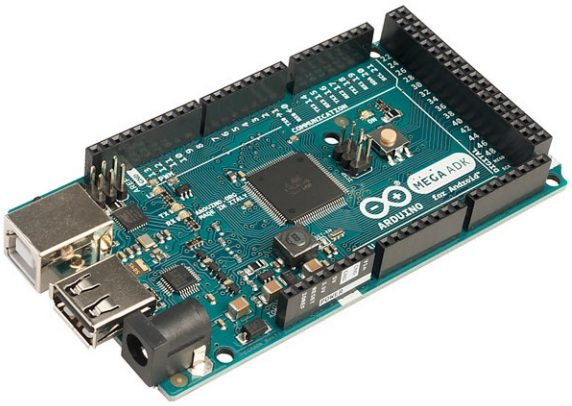


Figure 3.5: Arduino Mega ADK[4]

Although there are other boards that support the AOA protocol, as the IOIO board, the Arduino Mega ADK offers the possibility of executing tasks in the ATmega2560 microcontroller without any intervention of the Android device. This feature is very useful in case of future developments that include additional hardware to the smartphone-based quadrotor.

The Arduino Mega ADK is selected as the smartphone-to-ESC gateway. The workflow of this board is set as follows: the microcontroller receives a serial data frame from the smartphone through the USB port, each of the four PWM signal widths is extracted from the data frame, the PWM signals are sent to the ESCs, finally the cycle starts again. This workflow is executed each 2 ms and if the gateway does not receive any updated data frame from the smartphone, it keeps the last PWM signals set.

3.1.5 Battery

The quadrotor's battery must supply power to all the active components in the quadrotor. The maximum power consumption of these components, based on the nominal voltage value of a 3-cells lithium-ion polymer (LiPo) battery (11.1 V), is shown below.
The maximum current consumption in the quadrotor reaches 42.08 A, so the battery must be able to deliver current amplitudes greater than that value. A Floureon 3-cells LiPo battery with a nominal capacity of 4500 mAh and a nominal voltage of 11.1 V, was selected for this prototype and is shown in Fig. 3.6. This battery can deliver up to 30 times its nominal current, this is 135 A, continuously.

[4]Arduino Mega ADK image taken from https://goo.gl/ejeQeX

Table 3.2: Maximum Power Consumption of the Quadrotor's Components

Component	Voltage $[V]$	Max. Current $[A]$	Max. Power $[W]$
Smartphone	5	0.75	3.75
Arduino Mega ADK	11.1	0.75	8.33
4 x ESCs	11.1	1.38	15.32
4 x Motors	11.1	39.2	435.12

Figure 3.6: LiPo battery that powers the Quadrotor[5]

3.1.6 3D-printed Parts

The smartphone, the Arduino Mega ADK and the 4in1 ESC need to be easily placed and protected when installed on the frame. For that reason, multiple support components and one case were designed and 3D-printed using polylactic acid (PLA) filament. These 3D-printed objects are detailed below.

Smartphone Support

In order to place the smartphone near the *CoG* in the quadrotor and enable the possibility of capturing nadir photos or videos, the smartphone support is designed in such a way that it can be placed under the quadrotor's *CoG* but above the landing gear of the frame. Additionally, the support includes a free area that allows to use the complete field of view of the camera without being obstructed by it. The smartphone support is shown in Fig. 3.7.

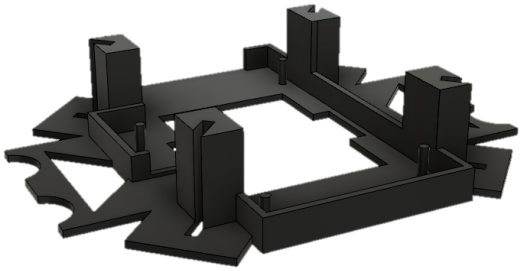
Arduino Mega ADK and ESC supports

The Arduino Mega ADK and the Emax 4in1 ESC are placed on the frame, above the quadrotor's *CoG*. Given the limited space available to locate these components

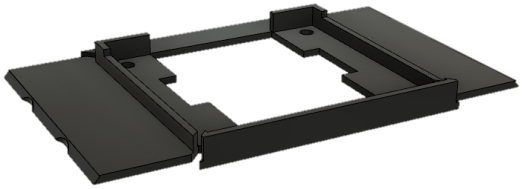
[5]Floureon 3S LiPo battery image taken from https://goo.gl/anC9M2

Figure 3.7: Smartphone support

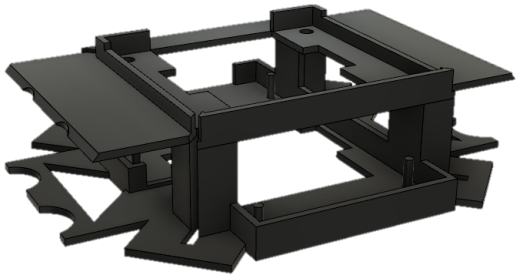
on the top of the frame, their supports are designed to be placed one on top of the other, as shown in Fig. 3.8.



(a) Arduino Mega ADK support



(b) ESCs support



(c) Arrangement of the Gateway and ESCs Support as a Whole

Figure 3.8: Arduino Mega ADK and EMAX 4in1 ESCs designed supports

Dome

To protect the Arduino Mega ADK and the ESCs, a dome that covers and encloses them is designed. This dome, shown in Fig. 3.9, totally encloses the Mega ADK and

ESCs supports restricting their movement and protecting them in case of a shock or hit.

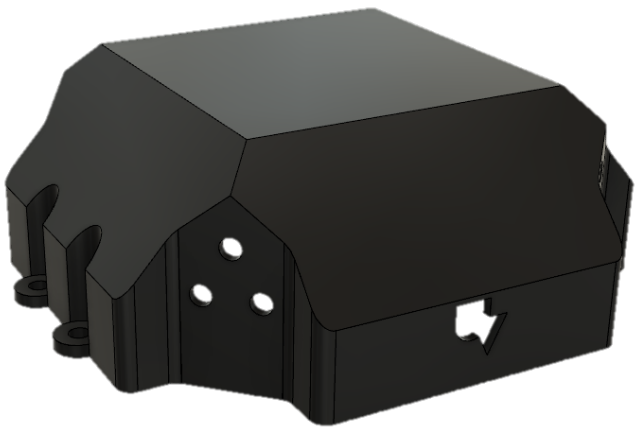


Figure 3.9: 3D Designed Dome

3.1.7 Assembled Smartphone-based Quadrotor

The smartphone-based quadrotor prototype is assembled using all the components exposed in this section, and is shown in Fig. 3.10.

Figure 3.10: Assembled Smartphone-based Quadrotor Prototype

3.2 Quadrotor Parameters

In this section, the quadrotor parameters are detailed. These parameters define the specific dynamic model for the smartphone-based quadrotor prototype and the conversion of control signals to correctly set the motors thrust.

3.2.1 Mass

The mass of each quadrotor's component is measured using a kitchen scale that has an uncertainty of ± 1 g. The results are shown in Table 3.3.

The total mass of the quadrotor m is then calculated as the sum of the weight of each of the components in the quadrotor, being $m = 1.568$ kg.

Table 3.3: Mass values of all the Quadrotor's components

Component	Mass $[kg]$
Smartphone	0.136
Frame	0.431
Battery	0.351
ESC	0.110
Arduino Mega ADK	0.330
Motors (4)	0.140
Propellers (4)	0.380
Smartphone support	0.105
ADK support	0.590
ESC support	0.350
Dome	0.130
Total	1.568

3.2.2 Moments of Inertia

As exposed by Lee et al. [2011], in a quadrotor, the moment of inertia matrix J is set as

$$J = \begin{bmatrix} J_{xx} & -J_{xy} & -J_{xz} \\ -J_{yx} & J_{yy} & -J_{yz} \\ -J_{zx} & -J_{zy} & J_{zz} \end{bmatrix}. \tag{3.1}$$

Taking into account that the quadrotor's symmetry with respect to the x and y axes is assumed, the J matrix can be approximated to

$$J \approx \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}. \tag{3.2}$$

The J_{xx}, J_{yy} and J_{zz} values can be obtained using multiple methods including: using a CAD model of the quadrotor and obtaining the inertia values from a 3D design software (Khodja et al. [2017]), approximating the shape of the quadrotor components to cylinders, cubes and other basic shapes to simplify the mathematical calculation of inertia (Jiřinec [2011]), and developing the bifilar pendulum experiment on the quadrotor (Garcia et al. [2017]). For this project, it was decided to obtain the quadrotor parameters experimentally, so the bifiliar pendulum experiment was developed.

In the bifilar pendulum experiment, an object is hung from two parallel ropes of

length r and separated by a distance $2l$, being allowed to rotate freely around a the axis that is parallel to the ropes. In Fig. 3.11, the geometry of the experiment to get J_{zz}, is shown. For the J_{xx} and J_{yy} inertias experiment, it is necessary to place the quadrotor hanging in such a way that the x and y axes are pointing paraller to the ropes, respectively.

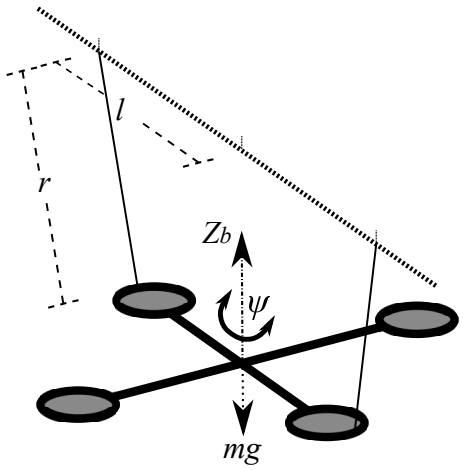


Figure 3.11: Bifilar pendulum experiment geometry for inertia identification

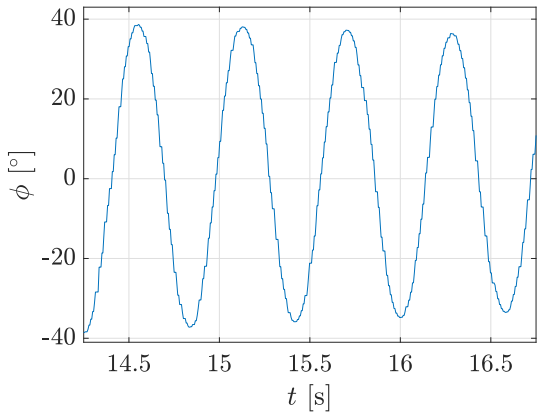
Once the quadrotor is hanging from the ropes, a small torque is applied manually to the quadrotor making it rotate about the vertical axis. The tension in the ropes makes the quadrotor swing, with a period of T_{osc} $[s]$, about the rotation axis.

The moment of inertia is then calculated using the bifiliar equation

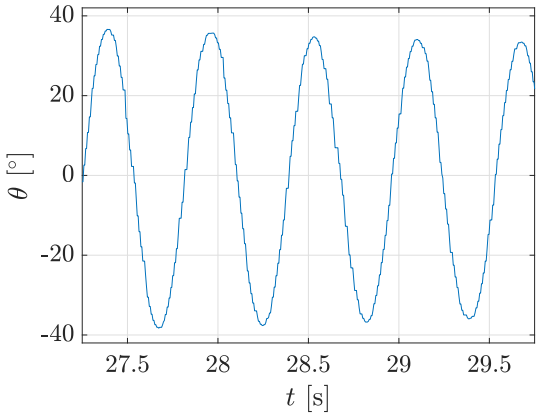
$$J.. = \frac{m|\vec{g}|T_{osc}^2 l^2}{4\pi^2 r} \ [kg \cdot m^2], \tag{3.3}$$

where $m = 1.568\ kg$ is the quadrotor's total mass and $|\vec{g}| = 9.807\ m/s^2$ is the gravity acceleration magnitude (Mustapa et al. [2016]).

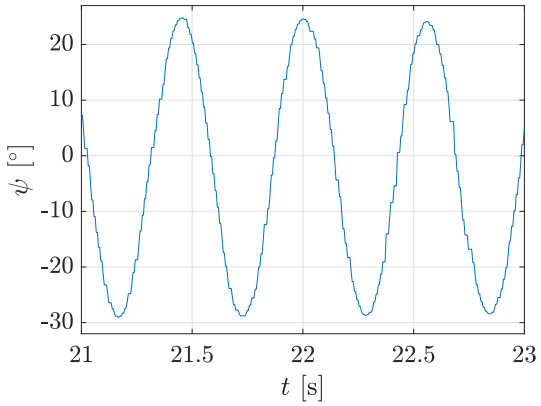
In Fig. 3.12, it is shown the excursion of the rotation angles, ϕ, θ and ψ, about the x, y and z axes respectively, while performing the bifilar pendulum experiment separately.



(a) Rotation about x axis, J_{xx} experiment



(b) Rotation about y axis, J_{yy} experiment



(c) Rotation about z axis, J_{zz} experiment

Figure 3.12: Rotation about x, y and z axes during the bifilar pendulum experiments

The resulting data got after the execution of the experiment around the three components of the quadrotor body frame, are shown in Table 3.4.

Table 3.4: Bifilar pendulum experiment results

Rotation Axis	r $[m]$	l $[m]$	T_{osc} $[s]$	Inertia value $[kg \cdot m^2]$
x	1.18	0.173	1.168	$J_{xx} = 0.0135$
y	1.10	0.173	1.080	$J_{yy} = 0.0124$
z	1.025	0.265	1.122	$J_{zz} = 0.0336$

3.2.3 Motors Thrust

As seen in Section 3.1, the motors rotational velocity ω_i is set by the ESC, which receives a PWM signal input. However, the inputs of the quadrotor (u, τ_ψ, τ_θ, and τ_ϕ) depend on the thrust force F_{M_i} applied by each quadrotor motor, as exposed in Section **??**.

In order to correctly set the desired force F_{M_i} in each motor during a flight, it is necessary to characterize the motor; and thus know how the force applied by the motor behaves with respect to the input PWM signal. This characterization is carried out by means of a thrust test.

In the thrust test, the motor and its corresponding propeller are fixed pointing up over a calibrated scale, as shown in Fig. 3.13, while connected to the ESC, and then the PWM signal is increased with steps of 10, with 0 being the minimum width of PWM signal and 255 the maximum, until reaching the maximum PWM width.

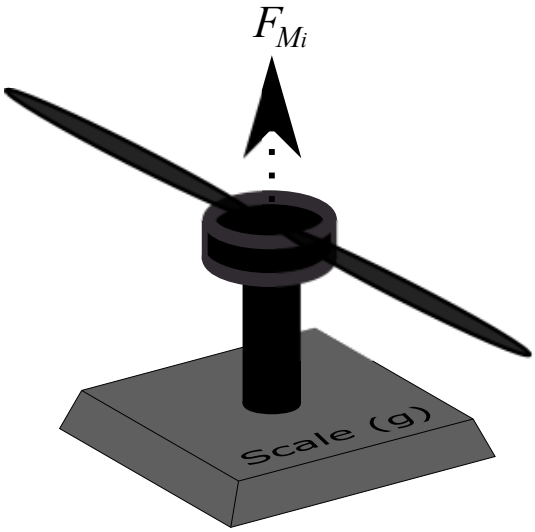
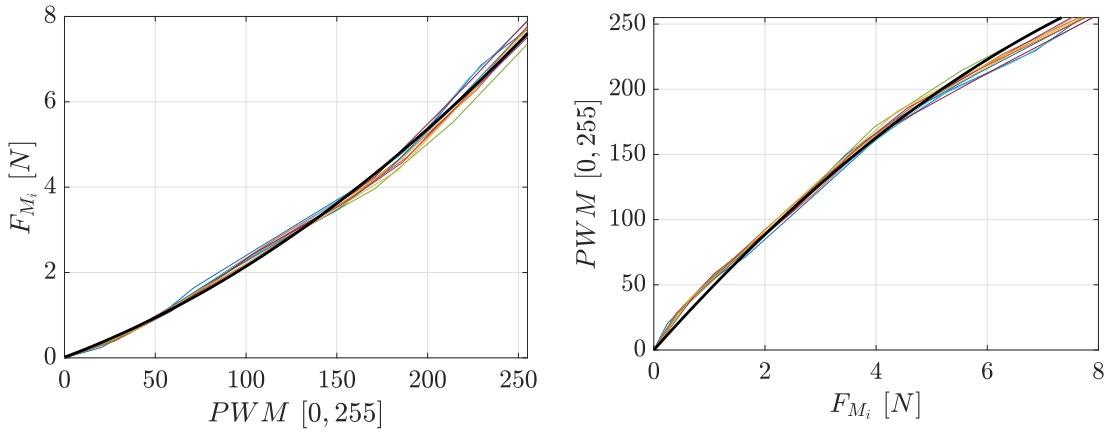


Figure 3.13: Thrust test configuration

With each PWM signal width increment, a scale reading is made. Since the readings m_s are obtained in units of mass (kg), the F_{M_i} must be calculated as

$$F_{M_i} = m_s|\vec{g}| \ [N]. \tag{3.4}$$

This test was developed twice with each motor and its results are shown in Fig. 3.14a.



(a) Motors characterization

(b) Inverse motors characterization

Figure 3.14: Motors thrust test results

In Fig. 3.14b, the inverse characterization is exposed. In this case, the PWM signal is defined as the dependent variable, and F_{M_i} as independent. This inverse characterization defines the PWM width setting that is sent from the smartphone to the Arduino Mega ADK.

The trend equations resulting from the thrust test are

$$F_{M_i} = (5.441 \times 10^{-5})(PWM)^2 + 0.01586(PWM) + 0.014808, \tag{3.5}$$

$$PWM = -1.983F_{M_i}^2 + 47.84F_{M_i} + 3.835, \tag{3.6}$$

where F_{M_i} is given in N, and PWM is a value between 0 and 255.

3.2.4 Motors Torque

The quadrotor suffers the application of a torque τ_{M_i} around the z axis when a motor M_i rotates. This torque affects the ψ angle indirectly by adding to the τ_ψ torque proportionally to the thrust force F_{M_i}, as

$$\tau_{M_i} = K_M F_{M_i}\ [N \cdot m]. \tag{3.7}$$

The torque τ_{M_i} is generated due to the conservation of momentum, and its unbalance is used to rotate the quadrotor about the z axis in the opposite direction to the rotation of the motor that generates it as exposed in **??**. In order to properly control the mentioned unbalance, it is necessary to find the value of the constant K_m.

The constant K_m can be found through a steady-state torque experiment, as stated by Costa De Oliveira [2011]. In this experiment, the z axis in the quadrotor is

located parallel to the ground, leaving the rotation about this axis as the only unblocked DoF. Using the geometry seen in Fig. 3.15, it is necessary to measure the force F_s that is generated when the motor M_i is rotating.

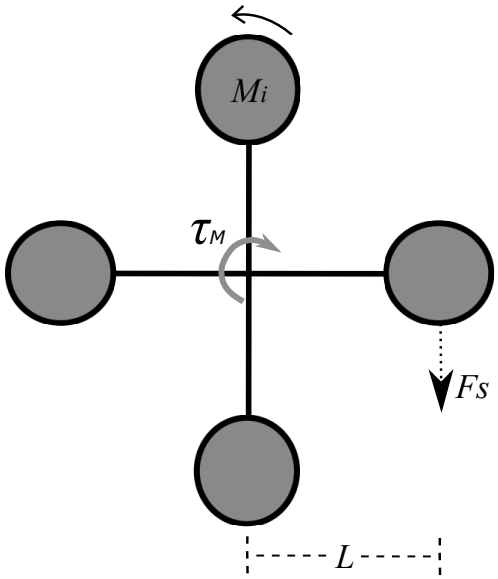


Figure 3.15: Motors torque experiment configuration

Taking into account that when the motor rotates clockwise, the quadrotor will tend to rotate counter-clockwise and vice versa, a scale is located to indirectly obtain the generated force F_s, using the reading of the weight m_s as $F_s = m_s|\vec{g}|\ [N]$. Using (3.5) and the PWM width steps used in the motors thrust experiment, multiple F_{M_i}-dependent F_s readings are obtained. The torques τ_M are then calculated by

$$\tau_M = F_s L\ [N \cdot m], \tag{3.8}$$

and their results are shown in Fig. 3.16.

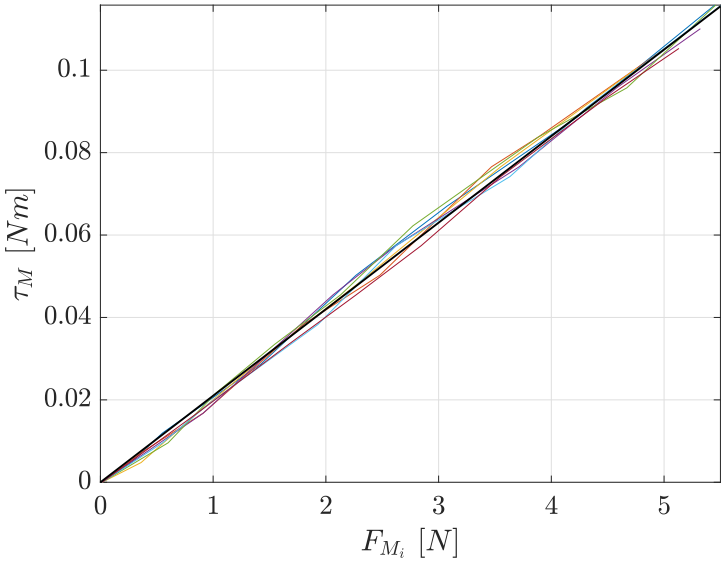


Figure 3.16: Torque

The relation between τ_M and F_{M_i} is linear as expected, and its slope defines the variable K_m as

$$K_m = 0.021\ [m]. \tag{3.9}$$

3.3 Conclusions

This chapter presented the physical composition and parametrization of the smartphone-based quadrotor prototype. In the first part, all the components of the quadrotor are detailed. A medium-size carbon fiber frame is used to support the complete quadrotor system. The LG Nexus 5X is selected as the smartphone where the control and estimation algorithms will be executed, which sends the control signals to an Arduino Mega ADK in order to generate PWM signals that set the motors rotational velocities through the ESC. These components are supported and protected by 3D-printed objects that are attached to the carbon fiber frame. On the other hand, the system is parameterized in its entirety using experimental methods. Here, the total mass and moments of inertia of the system are defined using a scale and the bifilar pendulum experiment respectively, while the motor's thrust and torque about the z-axis are found applying multiple PWM signals to the ESCs and obtaining the corresponding variable with the help of the scale.

Chapter 4

Control Strategies and State Estimation

rtrtrtrtr

4.1 Controllability and Observability

The controllability and observability features of the linearized system are checked using the controllability and observability Grammians W_c and W_o defined as

$$W_c = \int_0^\infty e^{At} B B^T e^{A^T t} dt, \tag{4.1}$$

$$W_o = \int_0^\infty e^{A^T t} C^T C e^{At} dt. \tag{4.2}$$

As the matrices W_c and W_o are positive definite matrices, then the system is both controllable and observable Werner [2012].

4.2 Control Strategies

This section exposes the controllers design procedure. Here, the mathematical procedure to design a linear quadratic Gaussian controller is described. It includes a regulator and an estimator, in addition to a gain compensator, allowing the system to track a trajectory. The design process of a H_∞ controller is shown, taking into account some weighting sensitivities.

4.2.1 Linear Quadratic Regulator

Designing an infinite-time regulator, a linear quadratic Gaussian (LQG) controller is set while looking for the minimization of the cost function V as

$$V = \int_0^\infty (X^T Q X + U^T R U) dt. \tag{4.3}$$

This minimization is achieved using Q and R as penalization matrices for the states and the inputs respectively, and the optimal input U^* is found as

$$\begin{aligned} U^* &= FX, \\ F &= -R^{-1} B^T P, \end{aligned} \tag{4.4}$$

such that P is a solution of the Algebraic Riccati Equation Werner [2013]

$$0 = PA + A^T P + Q - PBR^{-1}B^T P. \tag{4.5}$$

For this quadrotor controller, the tuning matrices Q and R were set as $Q = C^T C$ and $R = 0.7 * \mathcal{I}_{4\times4}$.

To find the linear quadratic estimator (LQE) gain F_e it is used the duality principle and then, F_e is defined as

$$F_e = -P_e C^T R_e^{-1} \tag{4.6}$$

such that P_e is a solution of the Filter Algebraic Riccati Equation

$$0 = P_e A^T + A P_e + Q_e - P_e C^T R_e^{-1} C P_e, \tag{4.7}$$

setting the tuning matrices as $Q_e = BB^T$ and $R_e = 0.9 * \mathcal{I}_{4\times4}$.

Thus, the LQG controller will be a mixture between the LQR and LQE where the closed-loop system is defined as

$$\begin{aligned} \dot{X} &= (A + BF + F_e C)X + BU, \\ Y &= CX, \end{aligned} \tag{4.8}$$

and is shown in Fig. 4.1.
As the LQR is not made for reference tracking, to let the quadrotor track a defined trajectory, it is necessary to include a gain filter v for the reference that compensates the total gain of the closed-loop system and enables the tracking of a reference input r. This gain filter is defined as

$$v = -(C * (A + BF + F_e C)^{-1} * B)^{-1}, \tag{4.9}$$

and its placing in the system is shown in Fig. 4.2.
where G_{cl} is the closed-system with the LQG controller described in (4.8).

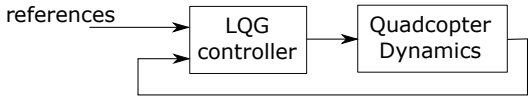


Figure 4.1: Closed-loop of the controlled system with an LQG controller.

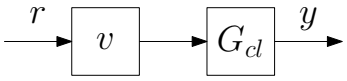


Figure 4.2: Closed-loop system with gain compensation for the LQG controller aiming to track a reference.

4.2.2 H_∞ Controller

For the design of the $H\infty$ controller, a generalized plant like the shown in the Fig. 4.3, was built.

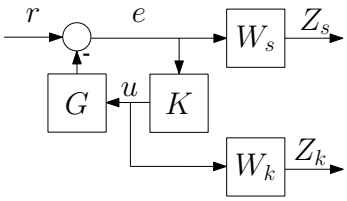


Figure 4.3: Generalized plant with the weighting filters W_s and W_k.

In this generalized plant, G represents the quadrotor dynamics, K the controller, r is the system reference, e is the error, u is the control input and W_s, W_k are weighting filters that must satisfy

$$\gamma = \left|\left| \begin{bmatrix} W_s S \\ W_k K S \end{bmatrix} \right|\right|_\infty < 1, \tag{4.10}$$

where S is the sensitivity function and KS is the control sensitivity defined as

$$S = (\mathcal{I} + GK)^{-1}, \quad KS = K(\mathcal{I} + GK)^{-1}. \tag{4.11}$$

The weighting filters for the sensitivity and the control sensitivity were chosen as

$$\begin{aligned} W_S &= \frac{w_s/M_s}{s + w_s} * \mathcal{I}_{4\times4} \ , \\ W_K &= \frac{c}{M_k} \frac{s + w_k}{s + c w_k} * \mathcal{I}_{4\times4} \ , \end{aligned} \tag{4.12}$$

where $w_s = 10^{-4}$, $M_s = 10^{-4}$, $w_k = 20$, $M_k = 20$, and $c = 10^3$. With these weighting filters, the value of γ is greater than one, and then it is necessary to rebuild the generalized plant using normalized weighting filters.

The H_∞ controller can be simulated as shown in Fig. 4.4.

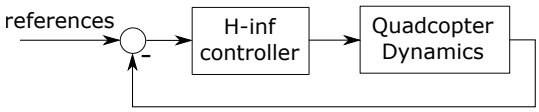


Figure 4.4: Closed-loop of the controlled system with an H_∞ controller.

H_∞ Controller Order Reduction

The designed H_∞ controller for our quadrotor, will have 4 outputs (the quadrotor has 4 control inputs), 4 inputs (the quadrotor has 4 measured outputs) and 20 states (due to the 12 order plant, 4 order W_k and 4 order W_s). To be able to implement this controller in a smartphone running Android, it it necessary to find a reduced order controller that behaves similarly to the full order controller K. To reduce the order of the controller, the singular values of the Hankel matrix

$$H_k = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{k-1} \end{bmatrix} \begin{bmatrix} B \\ AB \\ \vdots \\ A^{k-1}B \end{bmatrix}^T, \tag{4.13}$$

are analysed. The energy of the Hankel singular values is shown in Fig. 4.5.

As shown in Fig. 4.5, the last ordered four states have unnoticeable energy when it is plotted; that means that these four states can be truncated from the controller without modifying its dynamics Skogestad and Postlethwaite [2001].

4.3 Controllers Design

fhfgfhfh

4.3.1 Stabilize Mode

fghgfh

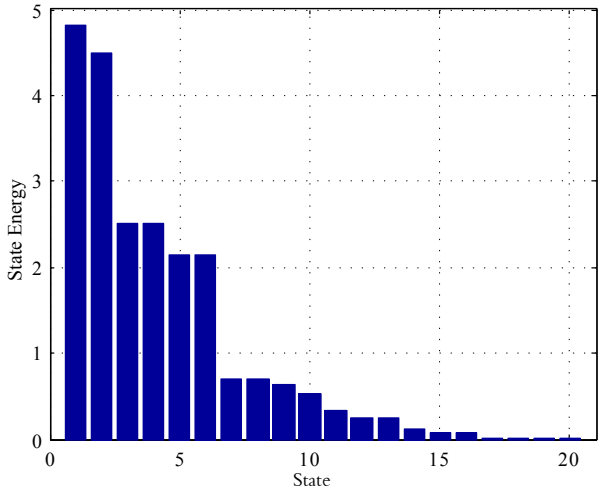


Figure 4.5: Hankel singular values energy histogram of the designed controller.

Dynamic Model

rtrterere

Linear Quadratic Regulator

rtrterere

H_∞ Controller

rtrtererre

4.3.2 Altitude Hold Mode

Dynamic Model

rtrterere

Linear Quadratic Regulator

rtrterere

H_∞ Controller

rtrtererre

4.3.3 GNSS Dependent Flight Modes

Dynamic Model

rtrterere

Linear Quadratic Regulator

rtrterere

H_∞ Controller

rtrtererre

4.4 State Estimation Through Kalman Filter

The quadrotor dynamics are sensed exclusively using the on-board smartphone's sensors. Considering that these sensors have different sample frequencies, and poor accuracy, it is necessary to use estimation algorithms, as a Kalman filter, to get reliable state data with constant sample frequency and improved accuracy and precision when compared with the raw data from the sensors.

4.4.1 Attitude Estimation

The Android API implements a Kalman filter for attitude estimation using the raw data deliverd by the smartphone's accelerometer, gyroscope and magnetometer, as exposed in Astudillo et al. [2017]. Using the quaternion Q_s delivered by the Rotation virtual sensor included in the Android SDK, it is obtained an absolute orientation representation with respect to the Earth frame Android Developer [2015], with

$$\begin{aligned} Q_s &= \mathrm{e}^{(\alpha/2)(u\vec{i}+v\vec{j}+w\vec{k})} \\ &= \begin{bmatrix} u\sin(\alpha/2) \\ v\sin(\alpha/2) \\ w\sin(\alpha/2) \\ \cos(\alpha/2) \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \end{aligned} \tag{4.14}$$

where α is the amount of degrees the quaternion is rotated around the axis $u\vec{i} + v\vec{j} + w\vec{k}$. The rotation matrix R_b^w can be defined using Q_s as

$$R_b^w = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_0q_2+q_1q_3) \\ 2(q_1q_2-q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_0q_1) \\ 2(q_1q_3+q_0q_2) & 2(q_0q_1+q_2q_3) & 1-2(q_1^2+q_2^2) \end{bmatrix}. \tag{4.15}$$

Comparing the terms in the two representations of R_b^w, the Euler angles are obtained as

$$\begin{bmatrix} \psi \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} atan2(2(q_3q_2+q_0q1), 1-2(q_1^2+q_2^2)) \\ arcsin(2(q_3q_1-q_2q_0)) \\ atan2(2(q_3q_0+q_1q2), 1-2(q_0^2+q_1^2)) \end{bmatrix}. \tag{4.16}$$

4.4.2 Position Estimation

Taking into account that the global navigation satellite systems (GNSS) receivers in smartphones have an accuracy of around 3 m and a sampling frequency of 1 Hz, a Kalman filter for position tracking is designed. In order to keep the estimation system independent from the application of controlling the quadrotor, this filter is based on the dynamics of a moving particle with constant acceleration between two samples

$$\xi(k+1) = \xi(k) + \dot{\xi}(k)t_k + 0.5\ddot{\xi}(k)t_k^2, \tag{4.17}$$

where t_k is the sample time.

Using $E_k = \begin{bmatrix} \xi & \dot{\xi} & \ddot{\xi} \end{bmatrix}^T = \begin{bmatrix} x_k & y_k & z_k & \dot{x}_k & \dot{y}_k & \dot{z}_k & \ddot{x}_k & \ddot{y}_k & \ddot{z}_k \end{bmatrix}^T$ as state vector and being

$$\Gamma = \begin{bmatrix} 1 & 0 & 0 & t_k & 0 & 0 & 0.5t_k^2 & 0 & 0 \\ 0 & 1 & 0 & 0 & t_k & 0 & 0 & 0.5t_k^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & t_k & 0 & 0 & t_k^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & t_k & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & t_k & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & t_k \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.18}$$

the matrix that satisfies $E_{k+1} = \Gamma E_k$, the prediction of the state $\hat{E}_k^-$ and its covariance P_k^- are obtained as

$$\hat{E}_k^- = \Gamma \hat{E}_{k-1}, \tag{4.19}$$

$$P_k^- = \Gamma P_{k-1} \Gamma^T + Q_k, \tag{4.20}$$

where $\hat{E}_{k-1}$ is the previous estimated state, P_{k-1} the previous error covariance matrix and Q_k the process variance. The state prediction is then corrected calculating the

Kalman gain vector K_k as

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}, \tag{4.21}$$

and updating the state estimation $\hat{E}_k$ and its covariance P_k, based on the measurements ζ_k as

$$\begin{aligned} \hat{E}_k &= \hat{E}_k^- + K_k(\zeta_k - H\hat{E}_k^-), \\ P_k &= (\mathcal{I} - K_k H)P_k^-, \end{aligned} \tag{4.22}$$

where R is the measurement covariance matrix, $\mathcal{I}$ is the identity matrix and H is the matrix that relate ζ_k and E_k. H is defined as

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.23}$$

considering that $\zeta_k \in R^6$ contains the GNSS measurements of x_m and y_m, the barometer measurements of z_m (in m), and the measurements of $\ddot{x}_m$, $\ddot{y}_m$ and $\ddot{z}_m$ in the Earth frame (in m/s^2).

The x_m and y_m position measurements are acquired using the GNSS receiver in the smartphone. Each sample of these coordinates are initially set in an ellipsoidal representation of decimal degrees by the receiver, and then converted to a bi-dimensional representation in meter units using the cartographic projection Magna-Sirgas.
The z_m measurements are acquired using the barometric pressure sensor which delivers the pressure value p_k in hPa. This signal is converted to meters as

$$z_m = 44330 \left(1 - \frac{p_k}{p_0}^{1/5.255}\right), \tag{4.24}$$

where p_0 is the atmospheric pressure at sea level Lauszus [2015].

The remaining measurement signals are the accelerations with respect to the Earth frame, $\ddot{\xi}_m$. These signals are calculated using the raw acceleration measurements from the smartphone a_b, which are represented with respect to the smartphone's body frame, and the attitude quaternion Q_s as

$$\ddot{\xi}_m = \begin{bmatrix} \ddot{x}_m & \ddot{y}_m & \ddot{z}_m \end{bmatrix}^T = Q_s a_b Q_s', \tag{4.25}$$

where Q_s' is the quaternion conjugate of Q_s.

The vector ζ_k is then set as

$$\zeta_k = \begin{bmatrix} x_m & y_m & z_m & \ddot{x}_m & \ddot{y}_m & \ddot{z}_m \end{bmatrix}^T. \tag{4.26}$$

4.4.3 Particle Model

rtrtrtrtrt

4.4.4 Quadrotor Model

ytytyttytt

4.5 Conclusions

This chapter presented the simulation and the testing of five control techniques for the attitude control of a quadrotor. The first technique is based on Lyapunov theory, it proved to be very reactive, especially for the yaw angle control. However, the stabilization in the direct neighborhood of the equilibrium point was not rigid enough to permit hover flight. The second one is a PID controller, it proved to be well adapted to the quadrotor when flying near hover. It was possible using this technique to successfully perform the first autonomous flight. The PID controller was only able to control the quadrotor in near hover and absence of large disturbances. The third one is an LQ controller, it displayed average stabilization results. It showed to be less dynamic than the PID. The fourth control technique is the Backstepping, its ability to control the orientation angles in presence of relatively high perturbations is very interesting. The sliding-mode technique is the fifth approach, it did not provide excellent results. The switching nature of the controller seems to be ill adapted to the dynamics of the quadrotor. The results of all these control approaches conducted to a combination of PID and Backstepping into the so-called Integral Backstepping. This was proposed as a single tool to design attitude, altitude and position controllers. The experiment has shown that OS4 is currently able to take-off, hover, land and avoid collisions automatically. As far as we know, OS4 is the first quadrotor practically capable of a collision avoidance maneuver.

Chapter 5

Implementation and Results

rgrtgrtgrgtrgrrg

5.1 Android Application

sdfdsfdsfds

5.2 Ground Control Station

sdfsdfdsd

5.3 Linear Quadratic Regulator Results

grtgrgtrgtrgtrg

5.3.1 Simple Translational Movements (LQR)

grtgtrgrtgrg

5.3.2 Trajectory Tracking (LQR)

trgrgrtgtrgr

5.4 H∞ Regulator Results

grgtrgrgtrg

5.4.1 Simple Translational Movements (H∞)

grtgrgrtgrtgtrg

5.4.2 Trajectory Tracking (H∞)

rgrtgrtgrgtrgrg

5.5 Conclusions

rgrgrgtrgrtgr

Conclusions and Outlook

In this thesis distributed algorithms

Publications

A. Astudillo, P. Muñoz, F. Alvarez and E. Rosero, "Altitude and attitude cascade controller for a smartphone-based quadcopter," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, jun 2017, pp. 1447–1454. [Online]. Available: http://ieeexplore.ieee.org/document/7991400/

A. Astudillo, B. Bacca and E. Rosero, "Optimal and robust controllers design for a smartphone-based quadrotor," in *2017 IEEE 3rd Colombian Conference on Automatic Control (CCAC)*

(Paper Submitted to Journal) A. Astudillo, P. Muñoz and E. Rosero, "Cascade Controller for Autonomous Flight of a Smartphone-based Quadrotor," in *Journal of Intelligent & Robotic Systems, SI: UAS-2017.*

Supplementary Material

All supplementary material including code, simulations, 3D objects designs, etc., is hosted in: https://goo.gl/U43bB6

Bibliography

Hao Liu and Xiafu Wang. Quaternion-based robust attitude control for quadrotors. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 920–925. IEEE, jun 2015. ISBN 978-1-4799-6010-1. doi: 10.1109/ICUAS.2015.7152379. URL http://ieeexplore.ieee.org/document/7152379/.

R Lopez, I. Gonzalez-Hernandez, S. Salazar, A. E. Rodriguez, J. J. Ordaz, and A. Osorio. Disturbance rejection for a Quadrotor aircraft through a robust control. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 409–415. IEEE, jun 2015. ISBN 978-1-4799-6010-1. doi: 10.1109/ICUAS.2015.7152317. URL http://ieeexplore.ieee.org/document/7152317/.

Jiwon Jung, Yeunduk Jung, Dongil You, and David Hyunchul Shim. A flight control system design for highly unstable unmanned combat aerial vehicles. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1117–1125. IEEE, may 2014. ISBN 978-1-4799-2376-2. doi: 10.1109/ICUAS.2014.6842365. URL http://ieeexplore.ieee.org/document/6842365/.

Satoshi Kohno and Kenji Uchiyama. Design of robust controller of fixed-wing UAV for transition flight. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1111–1116. IEEE, may 2014. ISBN 978-1-4799-2376-2. doi: 10.1109/ICUAS.2014.6842364. URL http://ieeexplore.ieee.org/document/6842364/.

Bo Shang, Jianxin Liu, Tiebiao Zhao, and Yangquan Chen. Fractional order robust visual servoing control of a quadrotor UAV with larger sampling period. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, volume 1, pages 1228–1234. IEEE, jun 2016. ISBN 978-1-4673-9334-8. doi: 10.1109/ICUAS.2016.7502645. URL http://ieeexplore.ieee.org/document/7502645/.

Sergio Salazar, Ivan Gonzalez-Hernandez, Ricardo Lopez, and Rogelio Lozano. Simulation and robust trajectory-tracking for a Quadrotor UAV. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1167–1174. IEEE, may 2014. ISBN 978-1-4799-2376-2. doi: 10.1109/ICUAS.2014.6842371. URL http://ieeexplore.ieee.org/document/6842371/.

Carolyn Pearce, Margaret Guckenberg, Bobby Holden, Andrew Leach, Ryan Hughes, Connie Xie, Meredith Hassett, Andrew Adderley, Laura E. Barnes, Mark Sherriff, and Gregory C. Lewin. Designing a spatially aware, automated quadcopter using an Android control system. In *2014 Systems and Information Engineering Design Symposium (SIEDS)*, volume 00, pages 23–28. IEEE, apr 2014. ISBN 978-1-4799-4836-9. doi: 10.1109/SIEDS.2014.6829921. URL http://ieeexplore.ieee.org/document/6829921/.

August Bjälemark and Hannes Bergkvist. Quadcopter control using Android based sensing. *Advances in Electrical and Computer Engineering*, pages 15–21, 2014.

Giuseppe Loianno, Gareth Cross, Chao Qu, Yash Mulgaonkar, Joel A Hesch, and Vijay Kumar. Flying Smartphones: Automated flight enabled by consumer electronics. *IEEE Robotics Automation Magazine*, Vol. 22:pages 24 – 32, 2015a.

Andrei Drumea. Control of Industrial Systems Using Android-Based Devices. *36th Int. Spring Seminar on Electronics Technology*, pages 405–408, 2013.

Kun-wei Lin, Zong-han Wu, and Yuan-li Lin. Applications of Temperature Control Based on Android Platform. *Proceedings of the International MultiConference of Engineers and Computer Scientists*, II, 2014. ISSN 20780958.

Nguyen-Vu Truong and Duc-Lung Vu. Remote monitoring and control of industrial process via wireless network and Android platform. *2012 International Conference on Control, Automation and Information Sciences (ICCAIS)*, (1):340–343, 2012. doi: 10.1109/ICCAIS.2012.6466614.

Leandro Florez Aristizabal, Diego F. Almario, and Jesus A. Lopez. Development of an Android App as a learning tool of dynamic systems and automatic control. In *2014 III International Congress of Engineering Mechatronics and Automation (CIIMA)*, pages 1–5. IEEE, oct 2014. ISBN 978-1-4799-7932-5. doi: 10.1109/CIIMA.2014.6983438. URL http://ieeexplore.ieee.org/document/6983438/.

Steven Wu Wu. *Sintonización de controladores PI/PID mediante el desarrollo de una aplicación en Android*. PhD thesis, Universidad de Costa Rica, 2013.

Jonathan García Téllez and Diego Fernando Ochoa Calambás. *Desarrollo de una plataforma de monitorización, control y comunicación con teléfonos inteligentes, para laboratorios portátiles de soporte al área de señales y sistemas*. Universidad del Valle, Cali, 2015.

Giuseppe Loianno, Yash Mulgaonkar, Chris Brunner, Dheeraj Ahuja, Arvind Ramanandan, Murali Chari, Serafin Diaz, and Vijay Kumar. Smartphones power flying robots. In *2015 IEEE/RSJ International Conference on Intelligent Robots*

and Systems (IROS), pages 1256–1263. IEEE, sep 2015b. ISBN 978-1-4799-9994-1. doi: 10.1109/IROS.2015.7353530. URL http://ieeexplore.ieee.org/document/7353530/.

Lorenzo Aldrovandi, Mohammad Hayajneh, Marco Melega, Michele Furci, Roberto Naldi, and Lorenzo Marconi. A smartphone based quadrotor: Attitude and position estimation. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1251–1259. IEEE, jun 2015. ISBN 978-1-4799-6010-1. doi: 10.1109/ICUAS.2015.7152418. URL http://ieeexplore.ieee.org/document/7152418/.

Phillip Bryant, Greg Gradwell, and David Claveau. Autonomous UAS controlled by onboard smartphone. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 451–454. IEEE, jun 2015. ISBN 978-1-4799-6010-1. doi: 10.1109/ICUAS.2015.7152322. URL http://ieeexplore.ieee.org/document/7152322/.

ArduPilot Dev Team. Copter - Flight Modes, 2016. URL http://ardupilot.org/copter/docs/flight-modes.html.

Tommaso Bresciani. *Modelling, Identification and Control of a Quadrotor Helicopter*. PhD thesis, Lund University, 2008.

Matthias Faessler, Davide Falanga, and Davide Scaramuzza. Thrust Mixing, Saturation, and Body-Rate Control for Accurate Aggressive Quadrotor Flight. *IEEE Robotics and Automation Letters*, PP (99):1–7, 2016. ISSN 2377-3766. doi: 10.1109/LRA.2016.2640362. URL http://rpg.ifi.uzh.ch/docs/RAL16{_}Faessler.pdf{%}0Ahttp://ieeexplore.ieee.org/document/7784809/.

M Emam and A Fakharian. Attitude tracking of quadrotor UAV via mixed H2/H∞ controller: An LMI based approach. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 390–395. IEEE, jun 2016. ISBN 978-1-4673-8345-5. doi: 10.1109/MED.2016.7535919. URL http://ieeexplore.ieee.org/document/7535919/.

Sherif Badr, Omar Mehrez, and A. E. Kabeel. A novel modification for a quadrotor design. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 702–710. IEEE, jun 2016. ISBN 978-1-4673-9334-8. doi: 10.1109/ICUAS.2016.7502536. URL http://ieeexplore.ieee.org/document/7502536/.

Samir Bouabdallah. Design and Control of Quadrotors With Application To Autonomous Flying. *École Polytechnique Fédérale De Lausanne, À La Faculté Des Sciences Et Techniques De L'Ingénieur*, 3727(3727):61, 2007. ISSN 0921-0296. doi: 10.5075/epfl-thesis-3727.

Francesco Sabatino. *Quadrotor control : modeling, nonlinear control design, and simulation*. Number June. KTH Royal Institute of Technology in Stockholm, Stockholm, Sweden, 2015.

Keun Uk Lee, Young Hun Yun, Wook Chang, Jin Bae Park, and Yoon Ho Choi. Modeling and Altitude Control of Quad-rotor UAV. In *Proceedings of International Conference on Control, Automation and Systems*, pages 1897–1902, 2011. ISBN 9788993215038.

Mohammed Abdallah Khodja, Mohamed Tadjine, Mohamed Seghir Boucherit, and Moussa Benzaoui. Experimental dynamics identification and control of a quadcopter. In *2017 6th International Conference on Systems and Control (ICSC)*, pages 498–502. IEEE, may 2017. ISBN 978-1-5090-3960-9. doi: 10.1109/ICoSC.2017.7958668. URL http://ieeexplore.ieee.org/document/7958668/.

Tomáš Jiřinec. *Stabilization and control of unmanned quadcopter*. Czech Technical University in Prague, Prague, Czech Republic, 2011. URL https://support.dce.felk.cvut.cz/mediawiki/images/d/d4/Dp{_}2011{_}jirinec{_}tomas.pdf.

Gonzalo A Garcia, A Ram Kim, Ethan Jackson, Shawn S. Kashmiri, and Daksh Shukla. Modeling and flight control of a commercial nano quadrotor. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 524–532. IEEE, jun 2017. ISBN 978-1-5090-4495-5. doi: 10.1109/ICUAS.2017.7991439. URL http://ieeexplore.ieee.org/document/7991439/.

Zaki Mustapa, Shakir Saat, A M Darsono, and H H Yusof. Experimental Validation of an Altitude Control for Quadcopter. *ARPN Journal of Engineering and Applied Sciences*, 11(6):3789–3795, 2016.

Marcelo De Lellis Costa De Oliveira. *Modeling, Identification and Control of a Quadrotor Aircraft*. Czech Technical University in Prague, jun 2011.

Herbert Werner. *Lecture Notes - Control Systems Theory and Design*. Hamburg University of Technology, Hamburg, 2012.

Herbert Werner. *Lecture Notes - Optimal and Robust Control*. Hamburg University of Technology, Hamburg, 2013.

S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design*, volume 2. John Wiley and Sons, 2001. ISBN 978-0470011683. doi: 978-0-470-01167-6.

Alejandro Astudillo, Pedro Munoz, Fredy Alvarez, and Esteban Rosero. Altitude and attitude cascade controller for a smartphone-based quadcopter. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1447–1454.

IEEE, jun 2017. ISBN 978-1-5090-4495-5. doi: 10.1109/ICUAS.2017.7991400. URL http://ieeexplore.ieee.org/document/7991400/.

Android Developer. SensorEvent, 2015. URL https://developer.android.com/.

Kristian Sloth Lauszus. *Flight Controller for Quad Rotor Helicopter in X-configuration*. PhD thesis, Kongens Lyngby, Denmark, 2015.