

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/292993131>

# Improvements of a flying-wing Micro-Air-Vehicle by closing the control-loop on ground

Technical Report · June 2014

DOI: 10.13140/RG.2.1.2070.8884

CITATIONS

0

READS

1,120

2 authors, including:



**Joshua Tristanco**

Universitat Politècnica de Catalunya

51 PUBLICATIONS 111 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Use of Ram Air Turbines for electrical taxiing in Airbus 320 [View project](#)



Innovative NDT technique based on ferrofluids for detection of surface cracks [View project](#)



Escola d'Enginyeria de Telecomunicació i  
Aeroespacial de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA

# TREBALL DE FI DE CARRERA

**TÍTOL DEL TFC:** Millores d'un petit vehicle aeri d'ala-volant tancant el llaç de control a terra

**TITULACIÓ:** Enginyeria Tècnica Aeronàutica, especialitat Aeronavegació

**AUTOR:** Irene Gallego Sánchez

**DIRECTOR:** Joshua Tristancho Martínez

**DATA:** 30 de Juny de 2014



**Título:** Mejoras de un pequeño vehículo aéreo de ala-volante cerrando el lazo de control en tierra

**Autor:** Irene Gallego Sánchez

**Director:** Joshua Tristancho Martínez

**Fecha:** 30 de Junio de 2014

## Resumen

Los Micro Air Vehicle (MAV) son pequeños vehículos volantes que debido a su reducido tamaño son muy útiles en aplicaciones "*indoor*" donde el uso de GPS está muy limitado. Una de las cargas de pago más usadas es el vídeo-enlace que, fáciles de embarcar independientemente del sistema de control del vehículo.

Dada una plataforma de vuelo MAV, el poner un piloto automático dentro de ella supone unas modificaciones importantes que encarecen el proceso y en algunos casos son prohibitivos debido a la fuerte limitación de peso. Una solución, en vez de embarcar un piloto automático, sería cerrar el lazo de control en el suelo usando el canal del control manual remoto del MAV. En vez de mandar el mando órdenes, sería la estación de tierra quien manda las órdenes emulando el control manual a partir de las imágenes. Para poder cerrar el lazo de control sería necesario mandar una telemetría adicional a través del canal audio del vídeo-enlace del MAV y hacer técnicas de procesado de imagen desde tierra para saber la posición y la actitud del MAV en tiempo real. Esta solución solo es viable para *indoor* que normalmente es el ámbito de trabajo de un MAV.

El uso de un cuatri-rotor como planta de potencia con plataforma giro-estabilizada simplifica enormemente la implementación del lazo de control desde tierra porque podemos separar en subsistemas el control del MAV. Al estar el piloto automático en tierra, la potencia de cálculo del no está condicionado a la limitación en peso tan fuerte del MAV cuando el piloto automático está a bordo. El estudiante implementará un subsistema de estabilización de altura del MAV que se pueda integrar en un futuro en el sistema ISIS de gestión de flotas de vehículos no tripulados del grupo ICARUS.

**Palabras clave:** Micro Air Vehicle, Indoor, Control loop, Inertial Measurement Unit, MAREA, ISIS, ICARUS



**Title:** Improvements of a flying-wing Micro-Air-Vehicle by closing the control-loop on ground

**Author:** Irene Gallego Sánchez

**Director:** Joshua Tristancho Martínez

**Date:** Juny, 30th 2014

## Overview

Micro Air Vehicles (MAVs) are a very small Unmanned Vehicles which provide interesting indoor implementations due to its reduced size. Therefore, on board payload is very restricted, so GPS use is very limited too. The most common payload is video-network and it works independently of vehicle control system.

In this context, the autopilot addition in a given MAV platform results in a significant changes that could raise the process and, in the worst case, could be too expensive because of the weight limitation. A proposed solution is to design an autopilot that close the control loop of the System “on ground” and links by the remote control channel. Consequently, the orders are sent from ground station as an emulation of the remote control. To close the control loop of the system would be necessary the telemetry acquisition and send it through audio channel from video-network onboard in the platform. In addition it would be necessary to process the images on ground station to know the attitude and relative position at real time. This is an only indoor solution which is the main MAV actuation environment.

Finally, a quad-rotor platform gyro-stabilized as a power plant simplifies immensely the closing of the control loop (in the ground station) implementation since we can divide the MAV control in different subsystems. Such the autopilot is on ground, it does not contribute to raise the weight restrictions, so the MAV's thrust is not compromised. It will be implemented a height stabilization subsystem in a MAV and could set up in ISIS system (fleet management) service by ICARUS group.

**Keywords:** Micro Air Vehicle, Indoor, Control loop, Inertial Measurement Unit, MAREA, ISIS, ICARUS



# INDEX

<b>ACRONYMS, ABBREVIATIONS AND DEFINITIONS .....</b>	<b>15</b>
<b>INTRODUCTION.....</b>	<b>17</b>
<b>CHAPTER 1. STATE OF THE ART .....</b>	<b>19</b>
<b>1.1 Control-loop for UAVs .....</b>	<b>19</b>
1.1.1 History of control.....	19
1.1.2 The control theory.....	20
1.1.3 Autopilot.....	23
<b>1.2 Relevant technologies.....</b>	<b>27</b>
1.2.1 Micro-Air-Vehicle .....	27
1.2.2 Flying wings .....	27
1.2.3 Arduino .....	28
1.2.4 UART data bus .....	28
<b>1.3 Similar solutions .....</b>	<b>29</b>
<b>CHAPTER 2. MAVS APPLICATIONS AND PAYLOADS.....</b>	<b>31</b>
<b>2.1 The inflatable flying-wing MAV specifications.....</b>	<b>31</b>
<b>2.2 Payload options .....</b>	<b>31</b>
<b>2.3 Flying-wing MAV performance .....</b>	<b>32</b>
2.3.1 Flight in Quad-rotor mode.....	32
2.3.2 Flight in flying wing mode .....	33
<b>2.4 Application fields .....</b>	<b>34</b>
2.4.1 Surveillance .....	34
2.4.2 Sensing.....	34
2.4.3 Management.....	35
2.4.4 Monitoring .....	35
2.4.5 Transport .....	35
<b>2.5 Required hardware .....</b>	<b>35</b>
<b>CHAPTER 3. MAV CONTROL-LOOP CLOSED ON GROUND .....</b>	<b>37</b>
<b>3.1 Control-loop architectures.....</b>	<b>37</b>
<b>3.2 Autopilot modes.....</b>	<b>38</b>
<b>3.3 Methodologies.....</b>	<b>40</b>
3.3.1 Control system as a whole .....	40
3.3.2 Image processing control .....	41
<b>3.4 User interface .....</b>	<b>42</b>
<b>CHAPTER 4. MAV HOLD HEIGHT MODE IMPLEMENTATION.....</b>	<b>43</b>
<b>4.1 Hardware implementation solution .....</b>	<b>43</b>



4.1.1	MAV platform .....	43
4.1.2	Remote control characterization.....	43
4.1.3	Remote control system implementation .....	47
4.1.4	Arduino microprocessor.....	48
4.1.5	PWM Requirements .....	51
4.1.6	Intercom protocol.....	53
4.2	<b>Software implementation solution .....</b>	<b>55</b>
4.2.1	Intercom protocol.....	55
4.2.2	Arduino programming .....	56
4.2.3	Labview programming .....	57
4.2.4	Remote piloting.....	61
4.2.5	Image recognition and processing .....	62
 <b>CHAPTER 5. CHARACTERIZATION OF THE NEW AUTOPILOT MODE...</b>		<b>65</b>
5.1	<b>Flying tests .....</b>	<b>65</b>
5.1.1	Initial system test.....	65
5.1.2	Remote control command test.....	65
5.1.3	Remote control virtual test.....	66
5.1.4	Autonomous height control tests .....	66
5.2	<b>Specifications of the system .....</b>	<b>67</b>
5.2.1	Remote control command tests results .....	67
5.2.2	Remote control virtual tests results .....	68
5.2.3	Remote control command test results .....	69
5.3	<b>Results analysis.....</b>	<b>70</b>
 <b>CHAPTER 6. CONCLUSIONS.....</b>		<b>71</b>
6.1	<b>General conclusions.....</b>	<b>71</b>
6.2	<b>Work conclusions .....</b>	<b>71</b>
6.3	<b>Future work.....</b>	<b>72</b>
6.4	<b>Environmental impact .....</b>	<b>72</b>
 <b>BIBLIOGRAPHY .....</b>		<b>75</b>

## LIST OF FIGURES

Figure 1 – Closed-loop control system .....	20
Figure 2 – Control Plant .....	21
Figure 3 – Control Plant diagram.....	21
Figure 4 – Closed-loop control system diagram .....	22
Figure 5 – Basic components of autopilot .....	24
Figure 6 – Displacement control system diagram.....	25
Figure 7 – Displacement autopilot with pitch rate feedback diagram.....	25
Figure 8 – Altitude hold mode of flight control system .....	25
Figure 9 – Flying-wing MAV prototype .....	27
Figure 10 – Transmitter and receiver serial data .....	28
Figure 11 – Quadrone propeller combination to generate various movements	31
Figure 12 – Command functions actuations of a) Throttle b) Roll c) Pitch d) Yaw .....	33
Figure 13 – Aircraft motion .....	33
Figure 14 – MAV's currently and proposed control system .....	37
Figure 15 – MAV controlled by virtual application.....	38
Figure 16 – Altitude hold mode of flight control system .....	39
Figure 17 – Detailed control system diagram .....	40
Figure 18 – Control system diagram for image processing function.....	41
Figure 19 – Proposed user interface .....	42
Figure 20 – Air platform, Quadrone mini .....	43
Figure 21 – Remote control front-end.....	44
Figure 22 – Frontal of the remote control inner .....	44
Figure 23 – Processor chip.....	45
Figure 24 – Radio system representation.....	46
Figure 25 – Processor and radio system integration .....	47
Figure 26 – Remote control design for control system (green boxes are the added components) .....	47
Figure 27 – New components addition (Arduino incorporation).....	48
Figure 28 – Arduino input connections to remote control commands (potentiometers). Yellow circle mark analog inputs from potentiometers. .	49
Figure 29 – Arduino connections.....	51
Figure 30 – PWM Signal in Arduino .....	51
Figure 31 – PWM Signal in the Oscilloscope.....	52
Figure 32 – First order filter – output PWM Arduino to radio input signal result	52
Figure 33 – PWM filter desired response .....	52
Figure 34 – Arduino and PWM conditioning implementation .....	53
Figure 35 – FTDI connections (Red board) .....	54
Figure 36 – Arduino placement inside the manual control and FTDI to USB ...	54
Figure 37 – Byte fragmentation .....	55
Figure 38 – Arduino code body .....	56
Figure 39 – Mask function .....	57
Figure 40 – Serial port communication Labview diagram (steps 1, 2 and 3) ...	58
Figure 41 – Virtual remote piloting Labview diagram.....	59
Figure 42 – Image processing Labview diagram .....	59
Figure 43 – Control unit Labview diagram .....	60
Figure 44 – Remote control mode function .....	61
Figure 45 – Remote control command test procedure .....	66

Figure 46 – Remote control virtual test procedure.....	66
Figure 47 – Control potentiometer Voltage vs digital value by Arduino .....	67
Figure 48 – Control potentiometer Voltage vs Total lift in grams.....	67
Figure 49 – Control virtually vs digital value in Arduino .....	68
Figure 50 – Control virtually vs total lift in grams .....	68
Figure 51 – Real time image while control process .....	69
Figure 52 – Under-damped response example .....	69
Figure 53 – Communication and command loop code diagram .....	84
Figure 54 – Image processing and autonomous control loop code diagram ....	85
Figure 55 – Complete user interface view .....	86
Figure 56 – Autonomous navigation detail user interface.....	86

## LIST OF TABLES

Table 1 – Command actions and MAV response depending on flight mode ....	34
Table 2 – Summary of hardware application .....	35
Table 3 – Command actions as a Quad and as a Flying wing.....	38
Table 4 – Movement responses as a Quad and as a Flying wing .....	39
Table 5 – Arduino connections .....	50
Table 6 – Byte ID corresponds with command .....	55
Table 7 – Original signal and result from Arduino processing .....	61
Table 8 – Image processing functions.....	63



## Acknowledgements

*Me gustaría dedicar el esfuerzo invertido en este Proyecto  
a las personas que de alguna manera han formado parte de él.*

*A Aymon por la conjunta colaboración en este Proyecto*

*A mi professor, Joshua Tristancho,  
que me ha enseñado a perseverar y a alcanzar.*

*A mis antiguos profesores, Oriol Busquets y Isidro Gutiérrez,  
Que crearon mi devoción por la ciencia.*

*A Marcos y a mi familia  
Por todo su apoyo*

*Gracias.*

## Trademarks

- $\dot{P}C$  is a trademark of PHILIPS
- WORD® is a trademark of Microsoft
- EXCEL® is a trademark of Microsoft
- Sparkfun is a registered trademark
- Wikimedia is registered by Wikimedia Commons



## ACRONYMS, ABBREVIATIONS AND DEFINITIONS

AP	Autopilot
CAS	Control Augmentation System
COM	Communication
FTDI	Future Technology Devices International
LED	Light Emitting Diode
MAV	Micro Air Vehicle
MIMO	Multi-Input and Multi-Output
OOP	Object Oriented Programming
PCB	Printed Circuit Board
PID	Proportional Integral and Derivative
PWM	Pulse Width Modulation
SAS	Stability Augmentation System
SCL	Serial Clock Line
SDL	Serial Data Line
SISO	Single-Input and Single-Output
UART	Universal Asynchronous Receiver/Transmitter
UAV	Unmanned Aerial Vehicle





## INTRODUCTION

Micro Air Vehicles (MAV) technology is, nowadays, rising and spreading in commercial sectors, mainly, as a toy. These platforms can be used for wide variety of applications, above all, indoor applications due to its characteristics.

Until now, there are designed and created a lot of MAV systems for specific implementations and with high complexity, but originating from new system.

This project proposes a system originating in a simple commercial system. It is selected a Quadrone commercial toy from **NINCOAIR** to study and develop some significant improvements and new implementations.

Quadrone Mini is a RF controlled MAV with four propellers and gyro-stabilized. With simple its structure, Quadrone allows easy manipulation.

In the work done before by **Aymon Padilla**, a flying-wing Micro Air Vehicle was developed to achieve one of many solutions, the result was the addition of an inflatable flying wing providing the MAV more endurance and a fifth propeller.

This study will take into account this new structure, but it will work with the initial prototype, in order to focus the project in the design and implementation of new control loop for the system and some other relating improvements. .

However, MAV platform entails strongly weight restrictions which will complicate the improvement processes, for this reason, it will be studied a control loop placed on ground to be implemented. This is a distinctive from the mostly existing MAV systems.

### Objectives

The project is divided into main objectives that must be accomplished and secondary objectives that shall be achieved but are not mandatory

Main objectives:

- Payload study of inflatable flying-wing Micro-Air-Vehicle to decrease the payload weight impact.
- Study of a commercial application for the MAV.
- Remote control emulation by computer (GS).
- Graphical interface design and implementation to control MAV as a remote control pilot.
- Image processing from camera at real-time to obtain MAV coordinates.

Secondary objectives:

- If possible export the described application to ISIS, from the ICARUS team, the UAS-ATM software
- Position (3D) interpolation from few led located in MAV

## **Written work distribution**

To achieve the objectives some simulations and studies will be performed like:

The first chapter is a state of the art about few topics like: Micro-Aerial Vehicles, Flying wings and above all, control theory. Also, the relevant technologies that takes part in the improvement processes.

The second chapter contains MAV's characterization from before study done, flying-wing MAV. From the obtained results, there is studied possible MAV applications related to commercial fields and the payload needed.

The third chapter is a design and prototype of the system control loop, supported in control theory described in first chapter and MAV specifications in chapter two.

The chapter four describes entirely the improvement implementations designed in chapter 3. Divided in two main parts, hardware implementations and software implementations.

The chapter five collect the tests definition to check the system functionality and finally there are exposed the results of these tests.

The chapter six has the conclusions, the future work and the environmental impact study.

Finally, there are the bibliography and the annexes.

## CHAPTER 1. STATE OF THE ART

The first chapter resumes the main themes tackle in the project. There will be introduced in control theory and especially in autopilots. Also, there are described relevant technologies needed for future improvements development.

### 1.1 Control-loop for UAVs

The automatic control has performed an essential role for the engineering development. Especially in aerospace vehicles, civil and military aviation, etc [1]. Modern aircraft include a variety of automatic control systems that aid the flight crew in navigation, flight management and it is essential for aircraft dynamics stability.

The meaning of control is to adjust to a requirement or to regulate<sup>1</sup>. In this context relating to engineering, the aim of a control theory is to calculate solutions for the proper corrective action from the controller that results in system stability. The system will hold the set point and not oscillate around it<sup>2</sup>.

#### 1.1.1 History of control

Despite control systems date back to antiquity, a more formal analysis began by **James Clerk Maxwell** in his paper “On Governors” in 1868. This explain the dynamics analysis of the centrifugal governor, using differential equations to describe the control system. This demonstrated the importance of mathematical models and methods and signalled the beginning of mathematical control and systems theory<sup>2</sup>.

In aviation area, the **Wright** brothers made their first successful test flights on December 17, 1903 and were distinguished by their ability to control their flights for substantial periods (more so than the ability to produce lift from an airfoil, which was known). Continuous, reliable control of the airplane was necessary for flights lasting longer than a few seconds<sup>2</sup>.

In the forties, the frequency response methods (specially the Bode diagrams) made possible for the engineers to design linear closed-loop control systems that satisfied performance requirements. After that, root-locus method proposed by **Evans** was fully developed [1].

Both, frequency response and root-locus methods, define classical control theory, which lead to stable systems to satisfy specific requirements.

The system analysis is carried out in complex-s domain using Laplace transform or in frequency domain by transforming from the complex-s domain. However, the scope of classical control theory is limited to single-input and single-output (SISO) system design<sup>3</sup>.

---

<sup>1</sup> <http://www.thefreedictionary.com/control> (February 2014)

<sup>2</sup> [http://en.wikipedia.org/wiki/Control\\_theory](http://en.wikipedia.org/wiki/Control_theory) (February 2014)

<sup>3</sup> [http://en.wikipedia.org/wiki/Control\\_engineering](http://en.wikipedia.org/wiki/Control_engineering) (February 2014)

Since 1960, because the ability of digital computers, introduced modern control theory, which is based on time-domain analysis of a physical system. Modern control theory made the design of control systems simpler because the theory is based on a model of an actual control system and can deal with multi-input and multi-output (MIMO) systems [1].

Modern control theory is carried out in the time domain using differential equations, as a set of decoupled first order differential equations defined using state variables<sup>3</sup>.

However, the system's stability is sensitive to the error between the actual system and its model (the system may not be stable). To avoid this situation, we design the control system by first setting up the range of possible errors and then designing the controller in such a way that, if the error of the system stays within the assumed range, the designed control system will stay stable. This method is called robust control theory and incorporates both the frequency response approach and the time-domain approach.

### 1.1.2 The control theory

Control systems may be thought of as having four functions: Measure, Compare, Compute, and Correct. These functions are completed by five elements: Detector, Transducer, Transmitter, Controller, and Final Control Element. The measuring function is completed by the detector, transducer and transmitter (Figure 1).

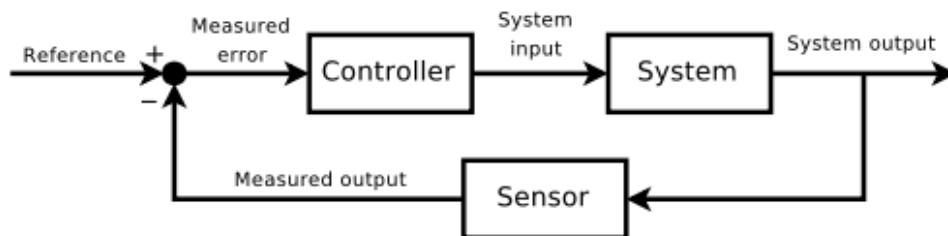


Figure 1 – Closed-loop control system

**The Control Plant** is the object of the control, it can be a single element or a set of elements (system). In this project we will consider the vehicle our plant to control.

The *controlled variables* are the variables used for the determination of the control goal of the plant. In the other hand, the *controlling variables* are the variables which can be changed or put from outside and which have impact on the *controlled variables*. Their values are the control decisions; the control is performed by the proper choosing and changing of these values. *Disturbances* are defined as the variables which except the controlling variables have impact on the controlled variables and characterize an influence of the environment on the plant. [2]

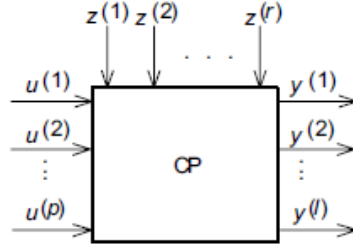


Figure 2 – Control Plant

**Closed-Loop Control Systems**, also called negative feed-back which has a stabilizing character.

In Figure 1 is represented a commonly closed-loop control system, where *controller* compares the actual value of the plant (*system*) output with the reference input (desired value), determines the deviation, and produces a control signal that will reduce the deviation to zero or to a small value; *sensor* is a device that converts the output variable into another suitable variable to compare to the reference input signal or desired value.

*Open-loop control systems* are systems in which the output has no effect on the control action, this means the output is not compared with the reference input.

**Transfer function**, also called system function, is commonly used to characterize the input-output relationships of system that can be described by linear, time-invariant, differential equations.

Mathematically, if differential equations are linear with constant coefficients, then, a transfer function can be obtained by taking their Laplace transform. If the differential equations are nonlinear and have a known solution, then, it may be possible to linearize and take their Laplace transform to obtain a transfer function.

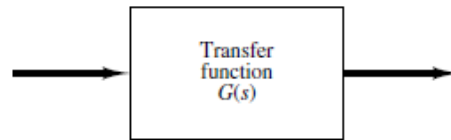


Figure 3 – Control Plant diagram

The Transfer function (1.1) , of the system in Figure 3, will be:

$$T(s) = G(s) = \frac{Y(s)}{X(s)} \quad Y(s) = G(s)X(s) \quad (1.1)$$

where  $X(s)$  is the Laplace transform of the input to the system and  $Y(s)$  is the Laplace transform of the output of the system, where we assume that all initial conditions involved are zero. In time domain:

$$y(t) = \int_0^t x(\tau)g(t - \tau)d\tau = \int_0^t g(\tau)x(t - \tau)d\tau \quad (1.2)$$

where  $g(t)$  and  $x(t)$  are 0 for  $t < 0$ .

The transfer function (1.3) in a closed-loop control system, Figure 4, will be:

$$T(s) = \frac{Y(s)}{X(s)} = \frac{C(s)}{R(s)} = \frac{G(s)}{1 + H(s)G(s)} \quad (1.3)$$

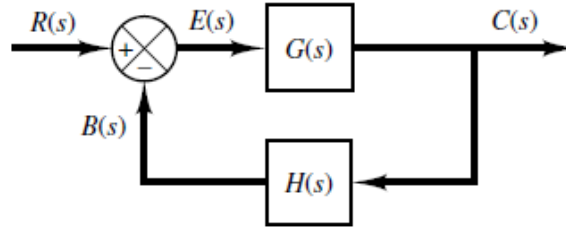


Figure 4 – Closed-loop control system diagram

Where:

$E(s)$  is the error signal

$R(s)$  is the reference input

$C(s)$  is the output signal

$B(s)$  is the feed-back signal

It is important to consider if the transfer function of a system is unknown, it may be established experimentally by introducing known inputs and studying the output of the system. Once established, a transfer function gives a full description of the dynamic characteristics of the system, as distinct from its physical description.

**Control actions** are the manner in which the automatic controller produces the control signal. We are following enumerated the most considerable control actions:

- Proportional control action: the relationship between the output of the controller  $u(t)$  and the actuating error signal  $e(t)$  is (1.4):

$$u(t) = K_p e(t) \quad (1.4)$$

which Laplace transform is (1.5):

$$\frac{U(s)}{E(s)} = K_p \quad (1.5)$$

where  $K_p$  is the proportional gain, the proportional controller is essentially an amplifier.

- Integral control action: the value of the controller output  $u(t)$  is changed at a rate proportional to the actuating error signal  $e(t)$ . That is (1.6):

$$\begin{aligned} \frac{du(t)}{dt} &= k_i e(t); \\ u(t) &= k_i \int_0^t e(t) dt \end{aligned} \quad (1.6)$$

The transfer function of the integral controller is (1.7):

$$\frac{U(s)}{E(s)} = \frac{K_i}{s} \quad (1.7)$$

where  $K_i$  is an adjustable constant.

- Derivative control action: the value of the controller output  $u(t)$  is changed at an inverse rate proportional to the actuating error signal  $e(t)$ . That is (1.8):

$$u(t) = k_d \frac{de(t)}{dt} \quad (1.8)$$

and the Laplace Transform is (1.9):

$$\frac{U(s)}{E(s)} = k_d s \quad (1.9)$$

where  $k_d$  is the proportional gain for derivative controller. It is always used in combination with proportional or proportional-integral control action.

- PID control (Proportional Integral and Derivative control): This combination has the advantages of each of the three individual control actions. The equation of a controller is given by (1.10):

$$u(t) = k_p e(t) + k_i \int_0^t e(t) dt + k_d \frac{de(t)}{dt} \quad (1.10)$$

or the transfer function (1.11):

$$\frac{U(s)}{E(s)} = k_p + \frac{k_i}{s} + k_d s \quad (1.11)$$

It is interesting to notice that more than half of the industrial controllers in use today are PID controllers or modified PID controllers.

### 1.1.3 Autopilot

The system to control the flight is called flight control system, which generally consists of three important parts.

- **The stability augmentation system (SAS):** augments to the stability of the aircraft. It mostly does this by using the control surfaces to make the aircraft more stable. A good example of a part of the SAS is the phugoid damper (or similarly, the yaw damper). The SAS is always on when the aircraft is flying. Without it, the aircraft is less stable or possibly even unstable.
- **The control augmentation system (CAS)** is a helpful tool for the pilot to control the aircraft. For example, the pilot can tell the CAS to 'keep the



current heading'. The CAS then follows this command. In this way, the pilot doesn't continuously have to compensate for heading changes himself. CAS are usually named as basic autopilots.

- **The Automatic control system or autopilot (AP)** takes things one step further. It automatically controls the trajectory of the vehicle. It does this by calculating (for example) the roll angles of the aircraft that are required to stay on a given flight path. It then makes sure that these roll angles are achieved. In this way, the airplane is controlled automatically.

Autopilots, in general - which can be mechanical, electrical, or hydraulic systems – were thought to assist the “human operator” in controlling the vehicle. Nowadays, autopilots have evolved significantly, from early autopilots that merely held an attitude to modern autopilots capable of performing automated landings under the supervision of a pilot<sup>4</sup>.

In Unmanned Air Vehicle (UAV), autopilot takes an important part of the system. Since UAV is an aircraft without a human pilot on board, the flight must to be controlled either autonomously by on board computers – autopilot – or remote piloting. Hence, flight control systems completely present at all phases of flight.

In autonomous control, the intelligence must come from sensors (gyroscopes, accelerometers, altimeters, airspeed indicators, automatic navigators, etc.). The autopilot supplies the necessary scale factors, dynamics (timing), and power to convert the sensor signals into control surface commands. These commands operate the normal aerodynamic controls of the vehicle [3].

Therefore, the advantages of autopilot over the remote pilot are in a MAV:

1. Autopilots have high reaction velocity as comparison to remote pilot.
2. They provide smooth control and have accurate knowledge of the aircraft state.
3. They can communicate well with computers, which is difficult for human.
4. They can execute multiple events and tasks at the same time.
5. Autopilot relieves human pilot from fatigue.

In Figure 5 is shown a diagram overview of an autopilot system [3]:

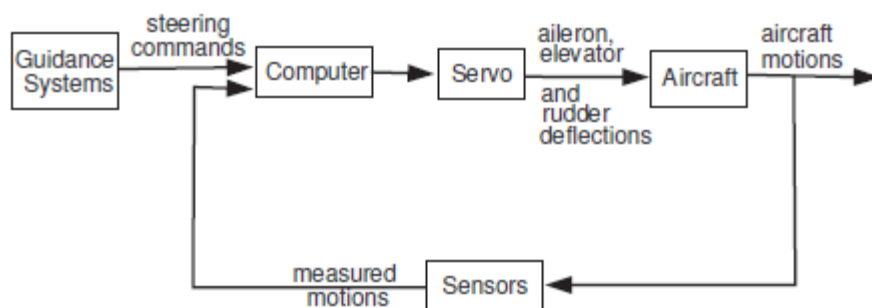


Figure 5 – Basic components of autopilot

<sup>4</sup> <http://en.wikipedia.org/wiki/Autopilot>

The basic aim of an autopilot is to track the desired goal, the basic MAVs autopilots are described below:

- **Displacement autopilot:** this is the first - and simplest – “autopilot” developed by Sperry Corporation. Its main function is holding the aircraft in straight and level flight with little or no manoeuvring capability, this means a correction in pitch attitude variations. Figure 6

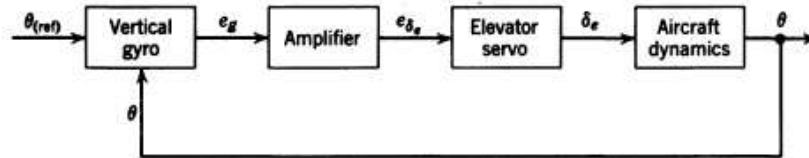


Figure 6 – Displacement control system diagram

This system uses the data from the vertical gyroscope and controls the aircraft through the elevators. The reference pitch angle  $\theta$  is thus the pitch angle that was present at the moment that the hold mode was activated.

It can be desired to increase the damping of the short-period oscillation by adding an inner feedback loop using a rate gyro (SAS inner loop) Figure 7.

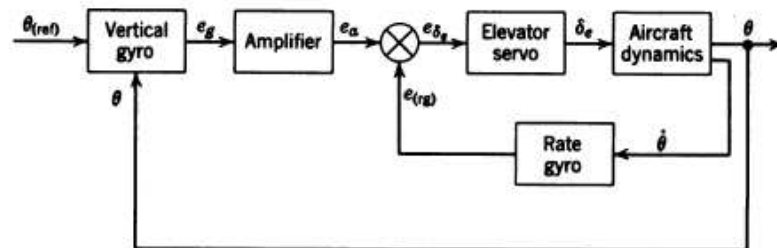


Figure 7 – Displacement autopilot with pitch rate feedback diagram

- **Altitude Autopilot:** which prevents pilots from constantly having to maintain their altitude. This is the same as controlling rate of climb at zero rate. Figure 8

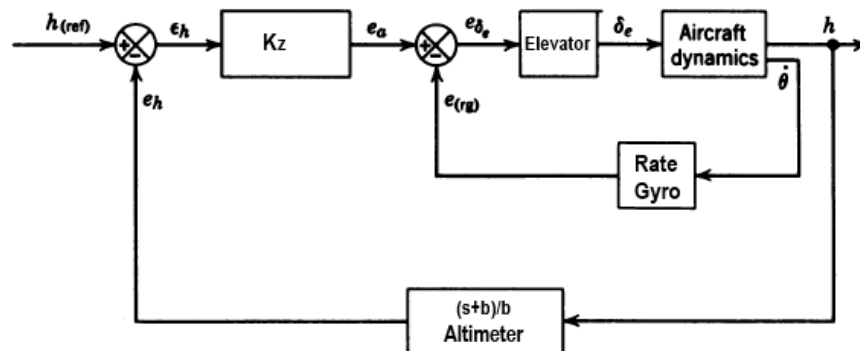


Figure 8 – Altitude hold mode of flight control system

The aircraft's flight path angle (and thus its altitude) is controlled by the elevators, and the airspeed or Mach number is controlled, either manually or automatically.

Altitude holding is one of the most important actuations for the future MAV applications.

## 1.2 Relevant technologies

### 1.2.1 Micro-Air-Vehicle

Micro Air Vehicle (MAV) is a class of unmanned air vehicle (UAV) with small dimensions. A MAV is the backbone of this project which has to be improved with an autonomous system.

MAVs are defined as smaller than 15 cm wing-span and very low weight, therefore its payload is very restricted. Consequently, MAVs are typically battery powered, hand launched and belly landed. MAVs have, also, short autonomy, but it depends on the system characteristics and, above all, its purpose.

All these restrictions cause interesting challenges designing autonomous modes of operation to determined purpose.

There are many types of MAV, depending on its dynamics (structure):

- Fixed-wing: The aircraft flights due to the vehicle's (wings) forward airspeed and generates lift force on the wings. The flight is straight.
- Rotary-wings: The lift force is generated by the aircraft's blades rotation. These vehicles are able to hover in a fixed position.
- Flapping-wings: This vehicle results as resemblance to nature, which flapping the wings is able to flight. Its mechanical dynamics is more complex than the other before.

### 1.2.2 Flying wings

A flying wing (Figure 9) is a tailless fixed-wing aircraft that has no definite fuselage, with most of the payload, and equipment being housed inside the main wing structure.

The advantage is, theoretically, the upper aerodynamic efficiency (lowest drag) design configuration for a fixed wing aircraft, this means structural efficiency and high energy efficiency.

Otherwise, because it lacks conventional stabilizing surfaces, in its purest form, the flying wing suffers from the inherent disadvantages of being unstable and difficult to control.



Figure 9 – Flying-wing MAV prototype

### 1.2.3 Arduino

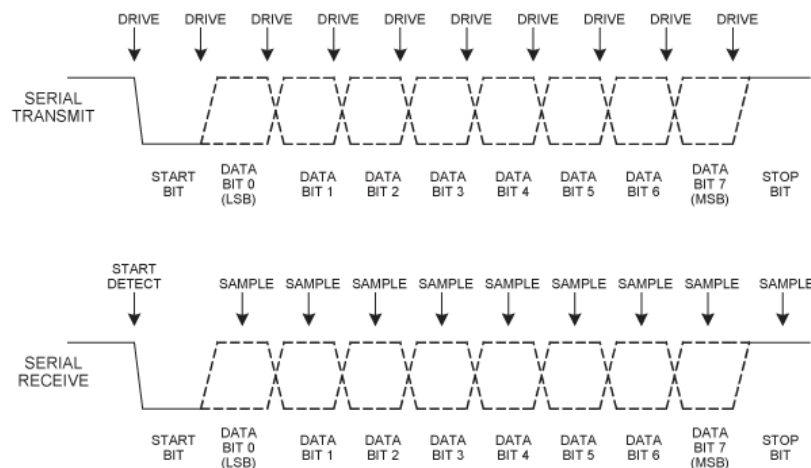
*Arduino* electronics prototyping platform is open-source. It is based on easy design hardware and software. It is intended for amateurs interested in develop interactive devices or platforms. The *Arduino Software* is a simple development platform having a serial connection through a USB port. The software is free and available in most Operative Systems. It was adopted by the community mainly in the academic environment but also in the professional business. There are lots of running examples than can be easily adapted to this project.

### 1.2.4 UART data bus

A universal asynchronous receiver/transmitter UART is usually an individual (or part of an integrated circuit) that translates data between parallel and serial forms. UART is used for serial communications, synchronous or asynchronous, over a computer or peripheral device serial port.

The UART takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. Each UART contains a shift register, which is the fundamental method of conversion between serial and parallel forms.

In character framing formation each character is sent as a logic low start bit, configurable number of data bits (usually 8), optional parity bit, and one or more logic high stop bits. Figure 10 shows the serial transmit and receive. UART operations are controlled by a clock signal which runs at a multiple of the data rate, typically 8 times the bit rate.



**Figure 10 – Transmitter and receiver serial data**

In application, transmitting and receiving UARTs must be set for the same bit speed, character length, parity, and stop bits for proper operation.

In addition to the basic work, a UART usually provide additional circuits for signals to indicate the state of the transmission media, and/or to regulate the flow of data.<sup>5</sup>

<sup>5</sup> [http://en.wikipedia.org/wiki/Universal\\_asynchronous\\_receiver/transmitter](http://en.wikipedia.org/wiki/Universal_asynchronous_receiver/transmitter)

## 1.3 Similar solutions

A list of similar solutions can be consulted in the following list of links:

<http://www.wdpsoftware.com/wdpsIRC2/index.html>

<http://winkleink.blogspot.co.uk/2013/02/raspberry-pi-gpio-scratch-remote.html>

<http://www.youtube.com/watch?v=oVWMH6qAvO4>

<https://www.youtube.com/watch?v=inHupXnqcnE>

[https://www.youtube.com/watch?v=EN\\_d3-3oww](https://www.youtube.com/watch?v=EN_d3-3oww)

<https://www.youtube.com/watch?v=TxoJstBXsl4>



## CHAPTER 2. MAVs APPLICATIONS AND PAYLOADS

The second chapter presents the possibilities of the studied platform and its applications in commercial fields. First, it is important the well-known of the system by specifications definition and characterization. Finally, it exposes the different improvements according to proposed applications and the needed payload. Taking into account MAV possibilities are in indoor environments.

### 2.1 The inflatable flying-wing MAV specifications

The final intention is to integrate the Quadrone power plant with an inflatable flying wing structure, but, at the initial point of the development we will base the study only in the Micro-air Vehicle (MAV). This quadrirotor is based on four propellers as can be seen in Figure 11. The combination of propeller speed results in various movements to be detailed later.

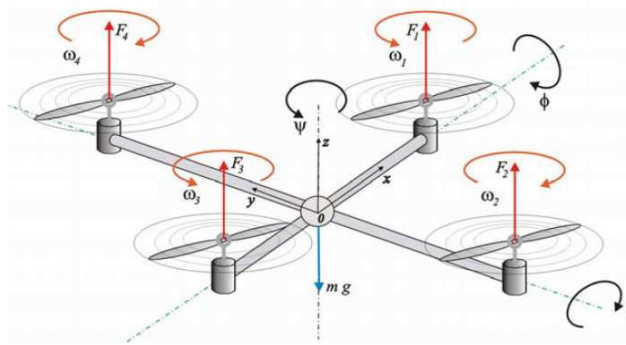


Figure 11 – Quadrone propeller combination to generate various movements

The power-plant platform is a Quadrone of 45 grams able to lift up to 15 grams. The endurance is more than 5 minutes but when the flying wing is integrated the endurance will be improved 4 times since only one propeller is required to maintain the flight.

### 2.2 Payload options

As there's been described in chapter before, initial MAV prototype has got only a stability augmentation system (SAS), which was designed and implemented by manufacturer.

Therefore, in order to improve the MAV control, it could be necessary to include additional payload to obtain flight data that let us to know the position, status, altitude, etc.

In addition, we will need a processor/computer (placed either on board or on ground) which compute the data and take a control decision.



This project proposes an entire on ground computer where the control action will take place. The way to achieve this goal will be analysed in the following chapters.

## 2.3 Flying-wing MAV performance

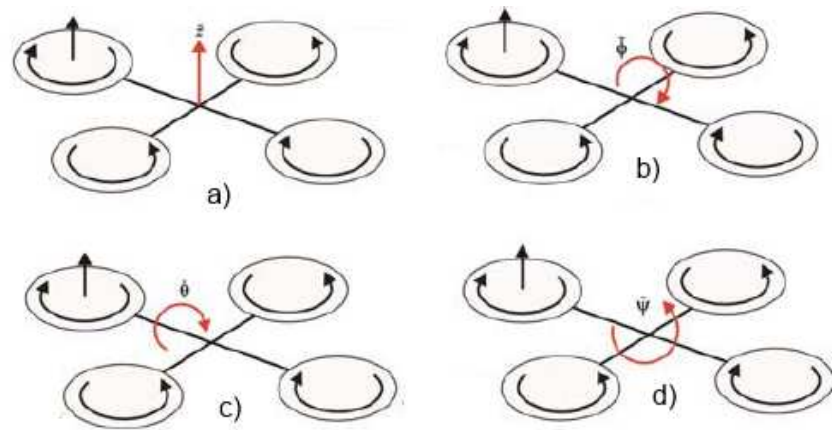
The MAV duality design supply a profitable flight performances, both quad-rotor and flying wing advantages. As a quad-rotor provides a vertical take-off, and easy directional control; in addition, as an aircraft provides extra endurance as a result of aerodynamic efficiency.

The following sections show the difference between flying wing mode dynamics and quad-rotor mode dynamics in response to the same command action.

### 2.3.1 Flight in Quad-rotor mode

The Quad-rotor mode responses depending on the command action are described below:

1. If *Throttle* command is pressed, Figure 12a, it is increased (or decreased) all the propeller speeds by the same amount. The quad-rotor responds with a vertical movement when it is in horizontal position.
2. When *Pitch* command is pressed, Figure 12c, this leads to torque with respect to the y axis which makes the quad-rotor turn. By doing so a pitch moment is created while keeping global thrust and torque unchanged. The quad-rotor will experience a translational movement, it will move forward with a negative pitch angle, and backwards with positive pitch angle. It should be noted that, as soon as the pitch angle change from the steady condition, the balance of forces will be broken and the quad-rotor will start to descend.
3. When *Roll* command is pressed, Figure 12b, the quad-rotor response is very similar to pitch action. This leads to torque with respect to the x axis, which makes the quad-rotor turn, as in the previous case, as soon as the roll starts, the quad-rotor will begin to descend.
4. Finally, if *Yaw* command is pressed, Figure 12d, the balance of torques is braked and the quad-rotor leads to torque with respect to the z axis which makes the quad-rotor turn. In this case the global thrust remains unchanged.



**Figure 12 – Command functions actuations of a) Throttle b) Roll c) Pitch d) Yaw**

### 2.3.2 Flight in flying wing mode

The MAV responses performed as a flying wing mode are more complex than before mode, since commands are thought for different dynamic, even so, the instantaneous responses are very similar. In this case is considered a continuous thrust force due to the 5<sup>th</sup> propeller.

1. If *Throttle* command is pressed, Figure 12a. The flying wing responds with a large vertical movement when it is in the horizontal position.
2. When *Pitch* command is pressed, Figure 12c, MAV experiments an opposed pair forces. A positive pitch angle induces a drag force (quad-rotor mode force) as well as a thrust force allowing the forward vertical displacement.
3. When *Roll* command is pressed, Figure 12b, the MAV torques with respect to the x axis which makes system turn and consequently, due to the unbalanced forces, generates a low spiral convergence.
4. If *Yaw* command is pressed, Figure 12d, affects similar as roll action, in this case, MAV torques with respect to the z axis, and after, endures a roll movement with slow vertical descend.



**Figure 13 – Aircraft motion**

On the whole, MAV performances are a combination between quad-rotor motion and aircraft motion. The first one have 4 separately motions as shown in Figure 12, and the second case have only 3 motion, pitch (which allow moving forward), and yaw and roll, that is directional and lateral control respectively.

The table below, Table 1, summarizes the different command actions and its effects in the MAV dynamics.

Command actions	Quad-rotor response	Flying wing response
<b>Throttle</b>	1. Vertical displacement	1. Large vertical displacement
<b>Pitch</b>	1. Pitch movement 2. Forward movement 3. Vertical movement	1. Pitch movement 2. Large vertical displacement
<b>Roll</b>	1. Roll movement 3. Descend	1. Roll movement 2. Turn 3. Descend
<b>Yaw</b>	1. Yaw movement	1. Yaw movement 2. Roll movement 3. Slow descend
<b>Transition Manoeuvre</b>	From Quad to Wing 1. Decrease pitch 2. Increase throttle 3. IAS increases 4. Increase pitch 5. Decrease throttle	From Flying wing to Quad 1. Throttle null 2. Large increase pitch 3. Full throttle 6. Ascend 5. IAS decrease 7. Descend 8. Pitch decrease

Table 1 – Command actions and MAV response depending on flight mode

## 2.4 Application fields

At this point, there have been exposed flying-wing MAV characteristics. Therefore, next step is to analyse feasible applications according to features exposed before. It is important to remark that **this system is useful in indoor actuations/missions**.

### 2.4.1 Surveillance

Indoor surveillance actuations, in buildings, warehouse or rooms. The MAV have to carry out a camera. Examples: Room inspections, people reconnaissance, etc.

### 2.4.2 Sensing

It could be necessary to detect some characteristics at indoor spaces. It will be needed specific sensor to each solution, for example, in a parking is wanted to know the air quality (CO<sub>2</sub> presence), it would needed a CO<sub>2</sub> sensor; or to detect the proper operation of a temperature conditioning system, it will be necessary a temperature sensor.

### 2.4.3 Management

Product management in a warehouse or building can be difficult. With a correct sensor it is possible to get data and then process it. For example, inventory actuation, scanning product (package) codes in high height shelves instead of human scanning.

### 2.4.4 Monitoring

Monitoring actuations are similar to surveillance; nevertheless, the aim is to record the scene. Therefore camera is a mandatory payload in the MAV. For example, difficult view recording in sport or buildings.

### 2.4.5 Transport

MAV is used to transport data information or voice information, physically, from site to site. In the contest, it would be necessary include a memory payload. Also, with proper performances, MAV would be able to carry light package.

## 2.5 Required hardware

Now, it will be considered the most significant applications with a flying-wing MAV and both on board and on ground payload needs.

**Basket recording** application is an example of monitoring based solution, with the aim to track rapidly the game in a basket match.

Other example is the **Auditorium temperature monitoring** application that supply a dynamic way to control efficiently the temperature in a room flying through all the volume inside the room. The Quadrone requires a temperature sensor.

The **Voice message indoor courier** is another example of application in digital data cases where indoor is shielded against radio-waves and wireless is not allowed. This application has the aim to carry digital information between two specific users that are separated inside the same building; information is physically sent from one user to other (i.e. confidential data)

Table 2 presents a summary of hardware applications for each example:

**Table 2 – Summary of hardware application**

Application	Camera	Modulator	Video-link	AP	IMU	GPS
Basket recording camera	HD camera	YES	YES	YES	YES	NO
Auditorium temperature monitoring	NO	Digital	Only TX	YES	NO	YES
Voice message indoor courier	Face recognition	YES	YES	YES	NO	NO



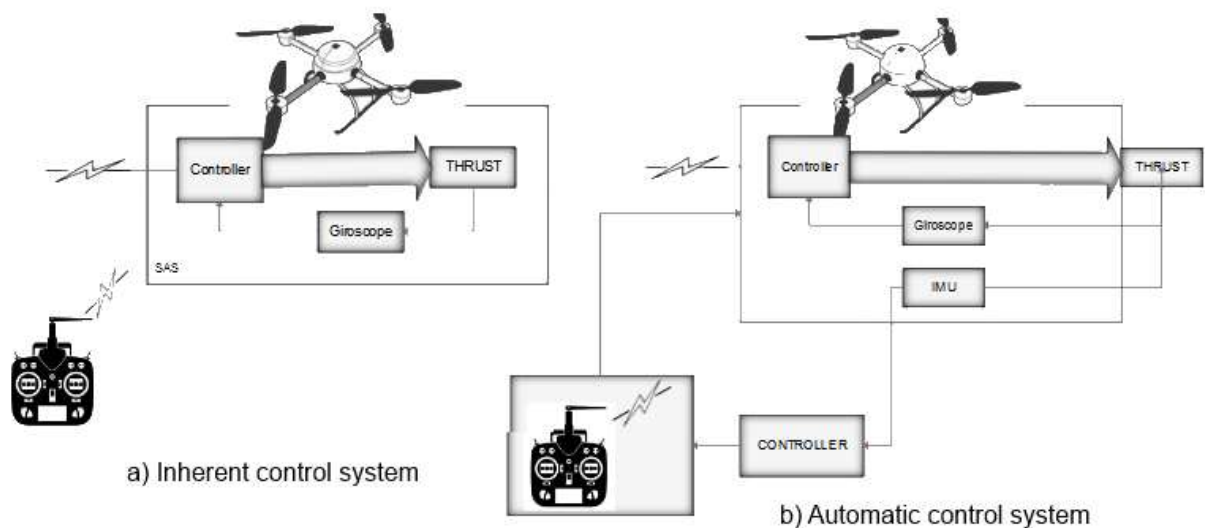
## CHAPTER 3. MAV CONTROL-LOOP CLOSED ON GROUND

The third chapter introduces the design and prototype to best implementation of the on ground control loop. There are compared on board with on ground control loop, supported in control theory described in chapter 1. Finally there is studied the control loop implementation in the Quadrone platform.

### 3.1 Control-loop architectures

As there has been described in first chapter, MAV control systems are based on closed-loop systems, which have a stabilizing character, this means, the system responds to input disturbances.

However, there are different methods to tackle the design. The initial MAV system has got integrated a SAS control-system (gyro-stabilizing system) Figure 14a. SAS actuates on the vehicle flight actuators to stabilize the platform in one or more axes automatically. Even so, this is not an autonomous navigation function, it will be necessary to implement an autopilot which introduces an outer feedback to acquire the control in specific navigation parameters. The design and implementation of the mentioned automatic control system are the main goal of the project. The system to be studied becomes as Figure 14b.



**Figure 14 – MAV's currently and proposed control system**

First, there is described the initial control system, this is how the system responds when controlled remotely. In this case, MAV is completely controlled by the pilot (with remote control). The command is picked with the joystick and transmitted to the platform which process the order and executes the command. This process is realized without any feedback. Ever so, a SAS control system is integrated to maintain the horizontal stability.

Since the hard payload restrictions, the elements to place on board are limited. For this reason, the new automatic control system (Figure 14b) is better placed on ground to not endanger MAV performances.

Consequently, this control system do not act direct in the MAV's actuation systems (on board control system), it will act through a radio system that could be just the remote command antenna.

Finally, to take advantage of the applied tools, it will be emulated the remote control in a virtual application. This will entail an agreement between remote and virtual orders Figure 15.

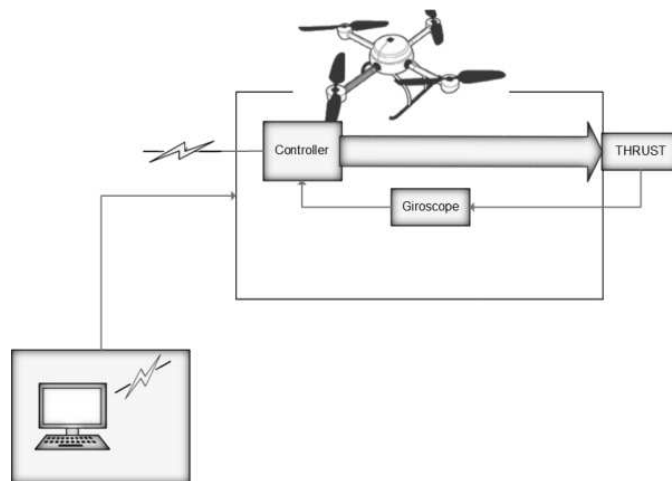


Figure 15 – MAV controlled by virtual application

This automatic navigation will bring the advantage to know exactly the command.

## 3.2 Autopilot modes

In Chapter 2 there has been described the flight modes responses considered in MAV's motions, resumed in the **¡Error! No se encuentra el origen de la referencia.** below:

Table 3 – Command actions as a Quad and as a Flying wing

Command actions	Quad-rotor response	Flying wing response
<b>Throttle</b>	1. Vertical displacement	1. Large vertical displacement
<b>Pitch</b>	1. Pitch movement 2. Forward movement 3. Vertical movement	1. Pitch movement 2. Large vertical displacement
<b>Roll</b>	1. Roll movement 3. Descend	1. Roll movement 2. Turn 3. Descend
<b>Yaw</b>	1. Yaw movement	1. Yaw movement 2. Roll movement 3. Slow descend

As there is shown, Quad-copter mode is simpler than flying wing mode, since the net movement effect in one axis only take place in that axis; that not happens when flying wing mode is on.

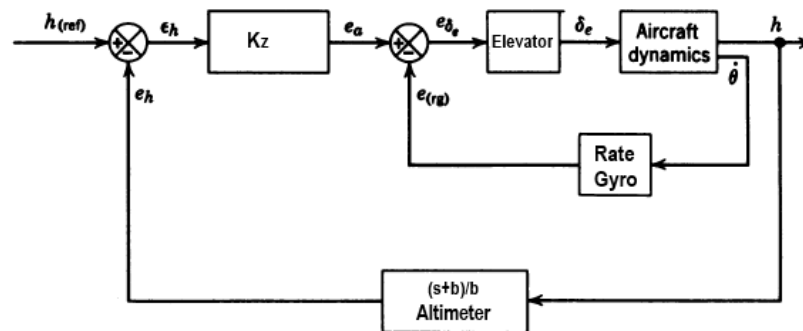
Now, it is important tackle the study in terms of automatic control, therefore, the input to study will be the axis, instead of the control command, and the effects for each mode.

**Table 4 – Movement responses as a Quad and as a Flying wing**

<b>Movement responses</b>	<b>Quad-rotor effects</b>	<b>Flying wing effects</b>
<b>Vertical axis (Throttle and Pitch)</b>	Vertical axis Throttle (simple) or pitch manoeuvre.	Vertical axis Pitch manoeuvre (involve an angle of attack).
<b>Lateral axis (Roll)</b>	Lateral axis Roll manoeuvre	Lateral and directional axis. Roll manoeuvre (involve roll angle) produces yaw moment.
<b>Directional axis (Yaw)</b>	Directional axis Yaw manoeuvre	Directional and lateral axis. Yaw manoeuvre (involve yaw angle) produces roll moment.

As a main goal of the project is to control MAV altitude, the axis to be controlled is the vertical axis, which is independent from the others.

Altitude Autopilot prevents pilot from constantly having to maintain the altitude. This is the same as controlling rate of climb at zero rate. See Figure 16.



**Figure 16 – Altitude hold mode of flight control system**

The future designed system will have an on ground video-camera sensor instead of an altimeter. The video-camera allows knowing MAV's position (thanks to the carried LED on MAV) due to the continuous image acquisition. The camera is placed in a determined point with a wide vision field. Later, the designed application will process the image to detect the blue LED (carried on MAV) at real-time, it is a tracking application .

To have the sensor on ground, connected wired to the control system, will make the system to respond faster and above all, favour the MAV performances.



### 3.3 Methodologies

#### 3.3.1 Control system as a whole

In order to finish the control system design, it is necessary to develop the detailed system diagram. There can be distinguished two main blocks: **MAV system** and **on ground system**. Now, there is compared the entire designed system relating to typical autopilot control system, shown in Chapter 1.2.

- MAV System is an imposed platform and it cannot experience a change. This system is related with aircraft and servos in the typical autopilot diagram. It will be considered as a black box since it is only known the input and the system response (output).
- On ground System will be the system developed, which corresponds to remaining systems in a typical autopilot. This system is composed by the remote command (the original remote link will become in a new one), the controller, the interface and the radio link.

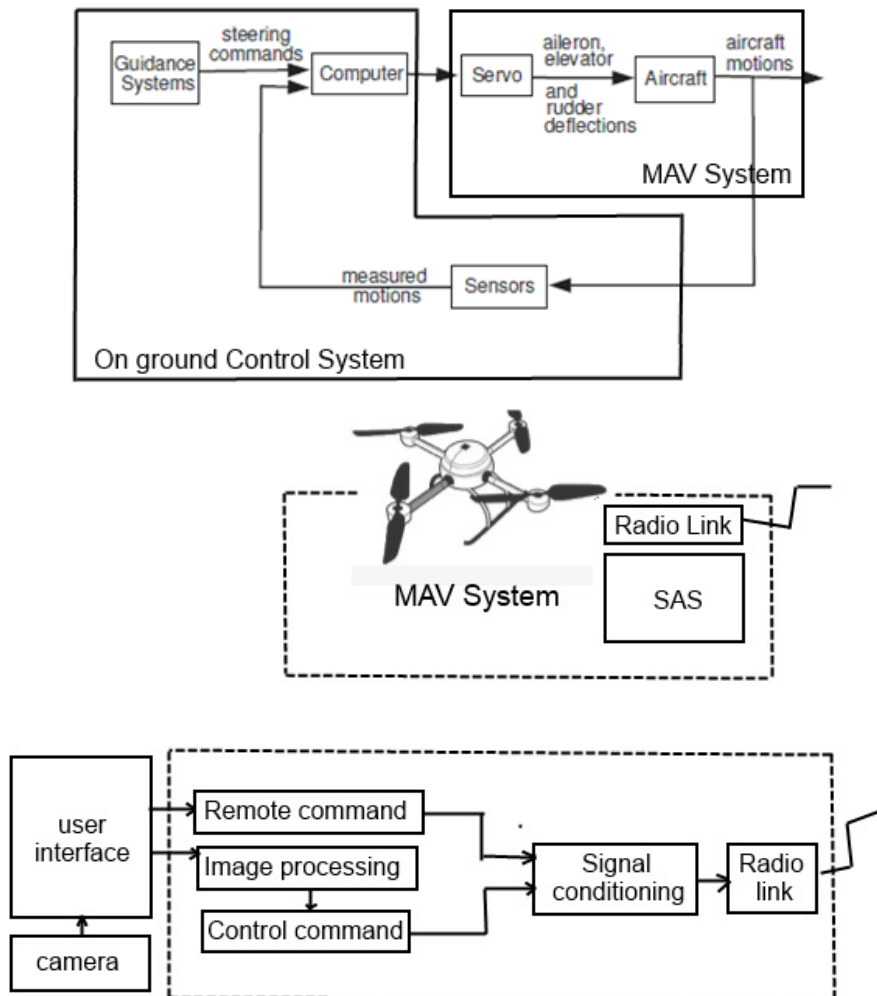


Figure 17 – Detailed control system diagram

Finally, the proposed control system is detailed in Figure 17 (taken into account system characteristics defined in the Section 3.1), and also the technology needed to perform the autopilot.

As mentioned in Chapter 3.2, the on ground control system will be able to operate both remotely piloted and in autopilot mode.

First case, remotely piloted can be executed either by the original remote-control, the system will work as the initial system; or by the virtual command, which is the remote-control emulated in the PC.

The second case, the navigation control (autopilot mode) is a most complex mode to develop.

The system designed have some differences relating with the autopilot system model that will affect as:

- The link between computer/controller and servomotors is neither wired nor direct; the developed system has to send the control order as it were a remote-control, through the radio link.
- The sensor/camera is, also, not connected to the MAV's output, since the sensor does not get MAV motion.

The on ground system (see Figure 17), on the whole, consists in:

1. Video camera: is the sensor system, one of the most important components which carries out the control.
2. Interface: which is the interaction element between the system and the user.
3. Processing: this bloc processes the data acquired either by sensor or by virtual command, in order to obtain a required value to transmit.
4. Signal conditioning: if is necessary to adapt the output signal from processing bloc to the radio input. It is an analog circuit.
5. Radio link allow the data transition from the on ground control system to the MAV system (as it was a commonly remote control command).

### 3.3.2 Image processing control

The ultimate function, which carries out the autonomous navigation, is image processing. Image processing function is in charge of compare the two states (desired state and currently state). The control diagram of this function (Figure 18)

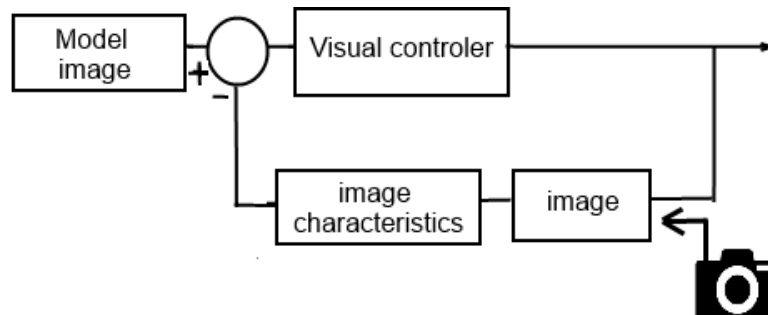


Figure 18 – Control system diagram for image processing function

### 3.4 User interface

The proposed user interface is shown in Figure 19. It is designed with the aim to create a simple interaction between user and application. It can be distinguished two parts: virtual remote control panel and autonomous navigation control panel, and, independently image panel.

- Virtual remote panel collect the 4 different commands (joystick) emulating the remote command. Each command can be pulled forward and backward
- Autonomous navigation panel shows the currently MAV height and allows the user set the desired height. When autonomous navigation button is activated control loop in navigation starts.

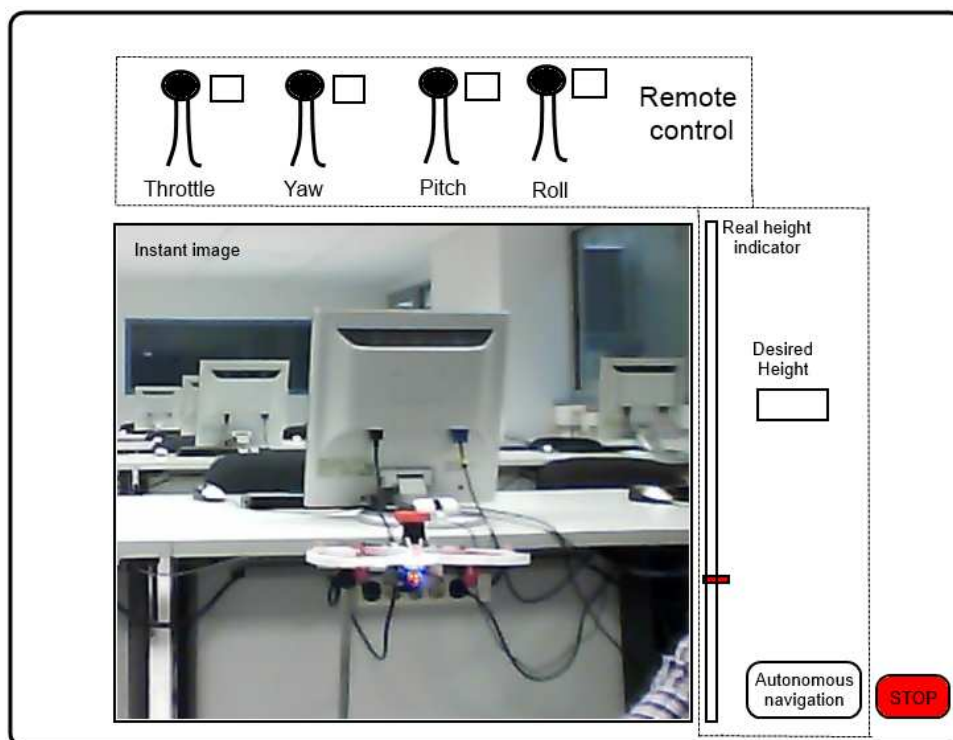


Figure 19 – Proposed user interface

## CHAPTER 4. MAV HOLD HEIGHT MODE IMPLEMENTATION

The fourth chapter describes the entire implementation, step by step, the on ground control loop, helped by annex documents. The description is divided in two parts, the hardware point of view and the software point of view. Moreover there are described other improvements in MAV relating with the control system implemented.

### 4.1 Hardware implementation solution

The first hardware implementation will be only developed on throttle command since the navigation control system, at the moment, will only takes part on this command. Therefore, next sections are described taking this rule into account.

#### 4.1.1 MAV platform

The air platform (Figure 20) has been described during the previous chapters in relation to their characteristics, performances, dynamics, etc. At this point, it is important to consider the platform as a black box, a group of systems where it is not possible to actuate. The platform, an imposed hardware, with a frequency radio signal as an input and dynamic response as an output.



Figure 20 – Air platform, Quadrone mini

#### 4.1.2 Remote control characterization

In the initial system, the remote control is in charge to transform the pilot actuation (joystick movements, Figure 21) into a control command, and finally sends the command to the MAV platform. In order to design this link, between the control system and the platform, it is necessary to understand how the remote control works, specially the radio system.



**Figure 21 – Remote control front-end**

Inside the manual control there is a PCB board with potentiometers, buttons, the processor and the radio transmitter module. Also there are three Trimmers for accurate control and the battery receptacle.



**Figure 22 – Frontal of the remote control inner**

Now, it can be distinguished the different remote control subsystems and its inputs and outputs (Figure 22). The most significant parts are described below:

- Input command (potentiometers): this subsystem allows pilot interaction through the remote control with MAV. Depending on the joystick position the signal value (in volts) varies, there are 4 different input command

signals corresponding with the MAV motions: yaw, throttle, pitch and roll. These values have to be processed before sent.

- Input trimmed: which allow trimming the input command signals. These signals only affect to the process phase.
- Chip BTLE 3372: this chip processes the input command signals and input trimmed signals, the outputs are connected to the radio system.
- Radio system: this system carries out the signal conditioning to be sent through the antenna. It consists in two subsystems, a chip and an antenna.
- Switcher: Switch on the remote control and connect to the MAV (radio interaction MAV-remote), and switch off.

#### Input command characterization:

Pin	Max value (Voltage)	Min value (Voltage)
Yin	3.3 V	0 V
Tin	3.3 V	0 V
Pin	3.3 V	0 V
Rin	3.3 V	0 V

#### Input trimmed characterization:

Pin	Max value (Voltage)	Min value (Voltage)
TY	3.3 V	0 V
TP	3.3 V	0 V
TR	3.3 V	0 V

#### Chip BTLE 3372 – processor characterization:

The Figure 23 represents the processor chip and its corresponding outputs and inputs. The input signals are Yin, Tin, Pin, Rin, TY, TP, TR and GND.

The output signals are LED, SPKR, P1, P2 (clock), P3, P3, P5 and P6 (Vcc).

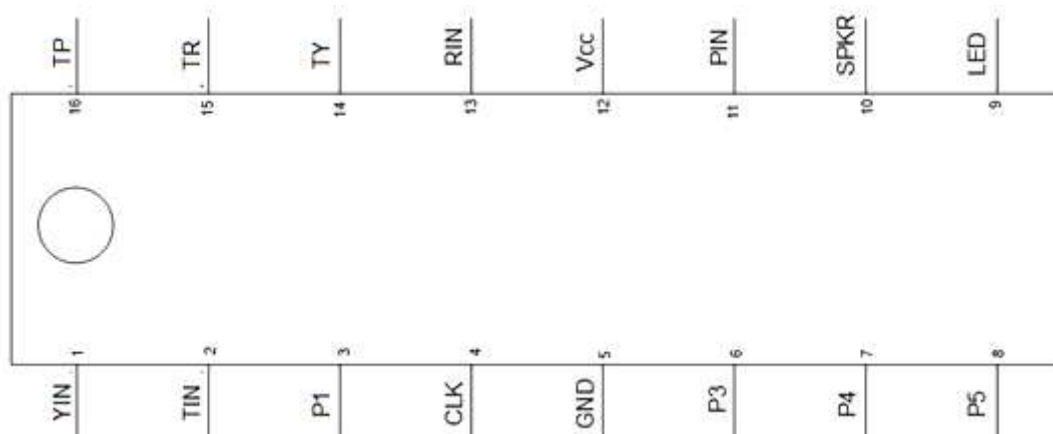


Figure 23 – Processor chip

The system values characterization are:

Pin	Max value	Min value	Measured value	Specifications
Yin (in)	3.3 V	0.00 V		Yaw input
Tin (in)	3.3 V	0.00 V		Throttle input
Pin (in)	3.3 V	0.00 V		Pitch input
Rin (in)	3.3 V	0.00 V		Roll input
TY (in)	3.3 V	0.00 V		Yam Trim input
TP (in)	3.3 V	0.00 V		Pitch trim input
TR (in)	3.3 V	0.00 V		Roll trim input
P1 (out)	3.3 V	0.00 V	2.58V	
P2 (out)	3.3 V	0.00 V	0.98V	CLK signal
P3 (out)	3.3 V	0.00 V	3.28V	
P4 (out)	3.3 V	0.00 V	0.08V	
P5 (out)	3.3 V	0.00 V	2.78V	
P6 (out)	3.3 V	0.00 V	3.25V	Vcc signal
GND (in)	3.3 V	0.00 V	0.00 V	GND
LED (out)	3.3 V	0.00 V	0.07 V	Signal led
SPKR (out)	3.3 V	0.00 V	0.00 V	Speaker signal

#### Radio subsystem characterization:

As Figure 24 shows, radio system's inputs are related to processor signal outputs. The discontinuous points represent the unknown part of the system, since there are not simple methods to characterize this zone.

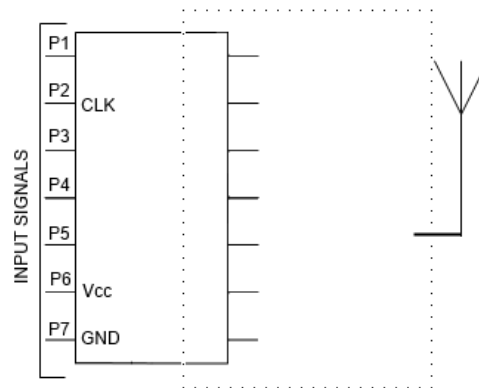


Figure 24 – Radio system representation

With the aim to characterize approximately the radio signal, it has been realized different tests with the oscilloscope with a non-expected results. The different signals registered are located around 2.4 GHz powerful signals.

Unfortunately, this is not enough to design and develop an easy radio prototype allowing the same link between the platform and the control unit. It would be needed data about codification process and modulation process which cannot be obtained through the previous tests.

Moreover, it has been not possible to obtain the information about both the radio system and the processor system, since there is any reference to relate these systems with corresponding commercial systems.

As a result, there is no fact which allows actuating between these systems two systems (radio processor and radio link) or emulates them in order to obtain the desired final result.

Finally, processor and radio system are considered again as a **black box** in the remote, known their inputs (input signals from joystick) and output (radio frequency signal).

**The advantage:** It is not necessary to perform a new signal processing system and antenna. **There will be developed the new control system actuating through the previous system**, the designed control system outputs have to be the same as the black box inputs (which are known) Figure 25.

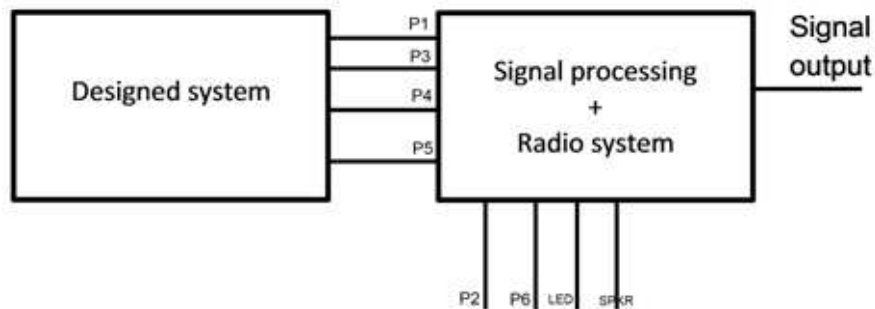


Figure 25 – Processor and radio system integration

From this point forward, this black box, which represents radio signal processor and radio system, will be considered the whole as a radio system.

#### 4.1.3 Remote control system implementation

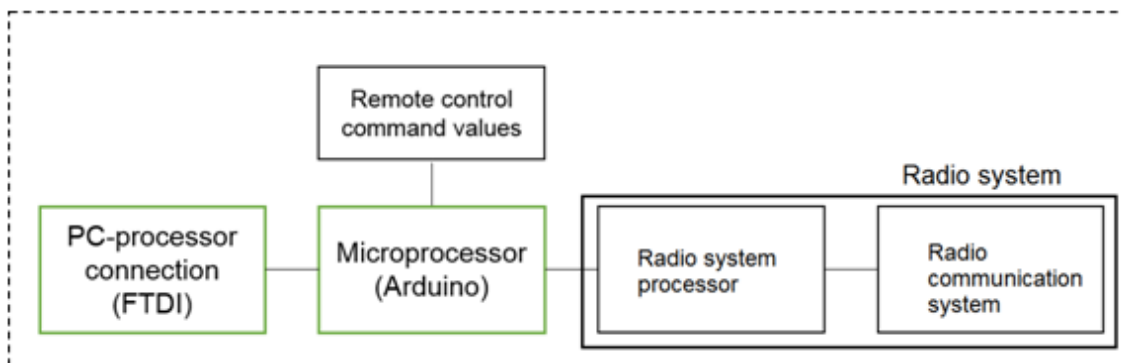


Figure 26 – Remote control design for control system (green boxes are the added components)

In Figure 26 the green boxes represent the components/systems added to work in agreement with the mentioned black box (radio system). Figure 27 shows the first steps in the new components addition.



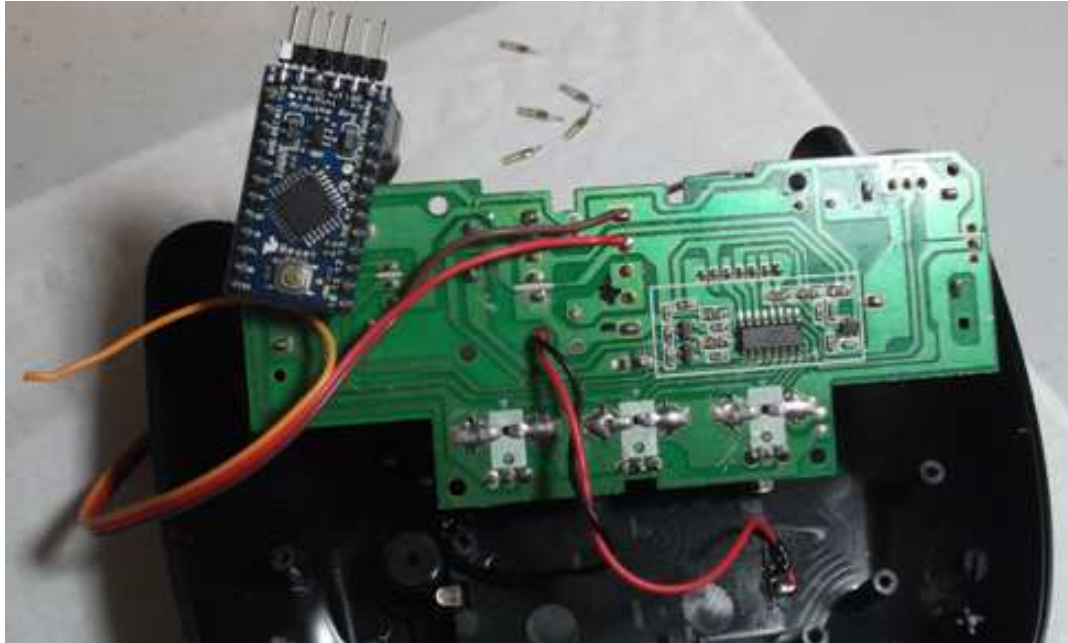


Figure 27 – New components addition (Arduino incorporation)

#### 4.1.4 Arduino microprocessor

An *Arduino Pro Mini* with the ATmega328 microprocessor is chosen to implement the system core (or the new system processor) to carry out the next functions:

- Signal acquisition
- Data interpreting (processing)
- Data sending and signal conditioning.

The microprocessor allows the control of the MAV, and also the digital interaction with a PC using a protocol to be defined in Section 4.2.1.

Since, the signals can come from both remote control commands and PC commands, the Arduino has to decide the signal to obey so it acts as a master.

The Arduino connections are enumerated and described in the **¡Error! No se encuentra el origen de la referencia.**, related to the signal source (see Figure 29).

Analog inputs from potentiometers can be wired directly to Arduino input PINS, since there are analog/digital converters inside the Arduino (see Figure 28).



**Figure 28 – Arduino input connections to remote control commands (potentiometers).  
Yellow circle mark analog inputs from potentiometers.**

Unfortunately, output signals from Arduino are an analog signal approximation, Pulse With Modulation (PWM) signal (Figure 30) which is not exactly the signal expected by radio system (discussed in the section below).

It is important to emphasize the designed conditioning system must have the same output signals' characteristics as the initial system when connected to radio system. Radio system input signals (corresponding with radio processor inputs) are analog ranged 0 to 3.3 V signals.

Connections	Source/ destination	Name	PIN	In Signal	OUT signal	Resolution
Analog inputs	remote control command signals	Tin, Yin Pin, Rin	A0, A1, A2 & A3	0V – 3.3V	Digital resolution 10 bit	Analog resolution 0.003V (input)
Analog outputs	Radio system (black box)	Tout, Yout, Pout, Rout	3, 5, 6 & 9	0V – 3.3V	8 bit	0.012V (output)
Digital connection (input and output)		Tx (trans mission) Rx (recept ion)		byte	byte	
Raw	battery			3.3 V		

**Table 5 – Arduino connections**

Leading this, it will be obtained the same MAV's behaviour. PWM signal conditioning process will be studied in next section.

Finally, to connect the Arduino to the PC is needed an FTDI, a chip that converts from RS232 to USB, in serial to serial, this chip will allow PC and Arduino interaction.

Regarding to power supplies, the system can be powered by two different sources, either FTDI chip or remote control battery. If remote piloting case, where is not necessary PC connection, the remote control is powered by inherent battery.

This double power supply can damage FTDI system when it is connected through USB and the battery is also feeding the board. To solve this problem, it

has been added a diode, not allowing current from FTDI to the battery when FTDI is connected but allowing battery feeds the Arduino board . See Figure 29a.

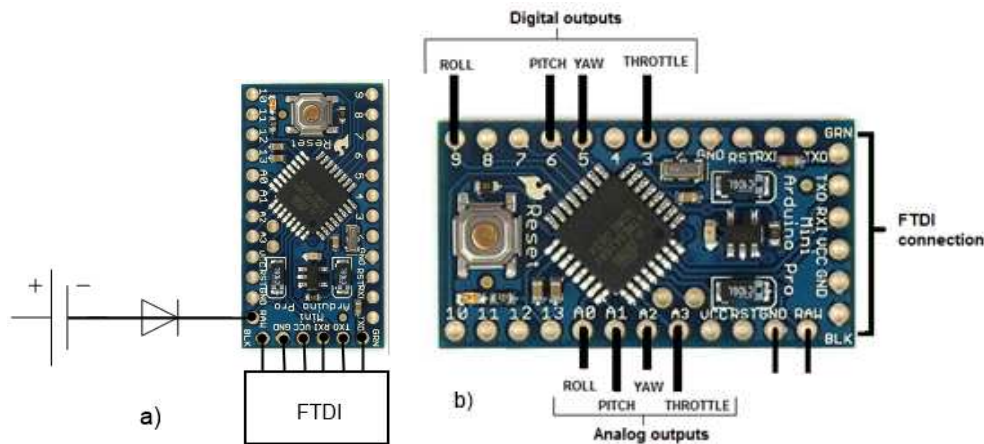


Figure 29 – Arduino connections

#### 4.1.5 PWM Requirements

PWM is a digital signal interpreted as an analog signal by the analog device. PWM technically consist in controlling the width of the pulse in a determined frequency. The on-off pattern (Figure 30) can simulate voltages between full on (3.3 Volts) and off (0 Volts) by changing average value of voltage (depends on the time the signal spends ON versus the time the signal spends OFF).

The PWM switching frequency has to be much faster than what would affect the device.

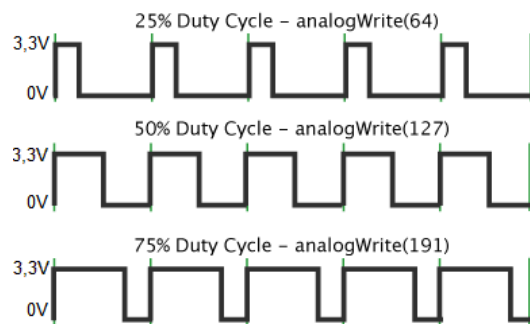


Figure 30 – PWM Signal in Arduino

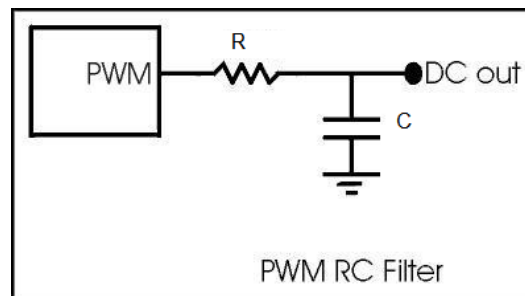
Although the device reads an analog signal, the rapid value changes make an abrupt device response.



**Figure 31 – PWM Signal in the Oscilloscope**

Consequently, PWM signal has to be treated in order to be well understood by radio system. To convert a PWM signal to an analog signal it is necessary to implement a low pass filter. This filter will attenuate the frequencies higher than cut-off frequency, so the abrupt changes in PWM will be reduced.

It is chosen a first order filter as Figure 32 shows.

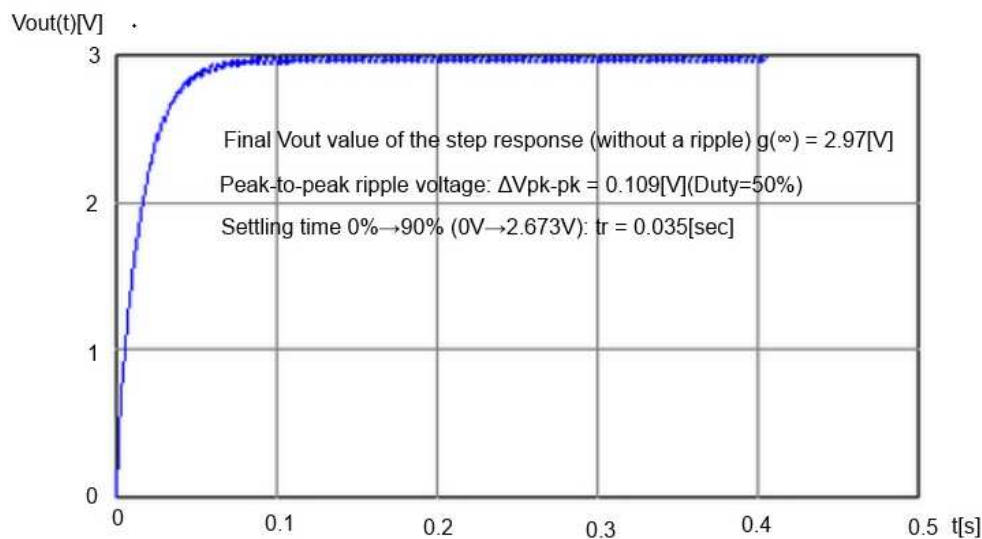


**Figure 32 – First order filter – output PWM Arduino to radio input signal result**

It has to be considered the settling time and the ripple voltage in order to best design and it is an agreement between R value and C value.

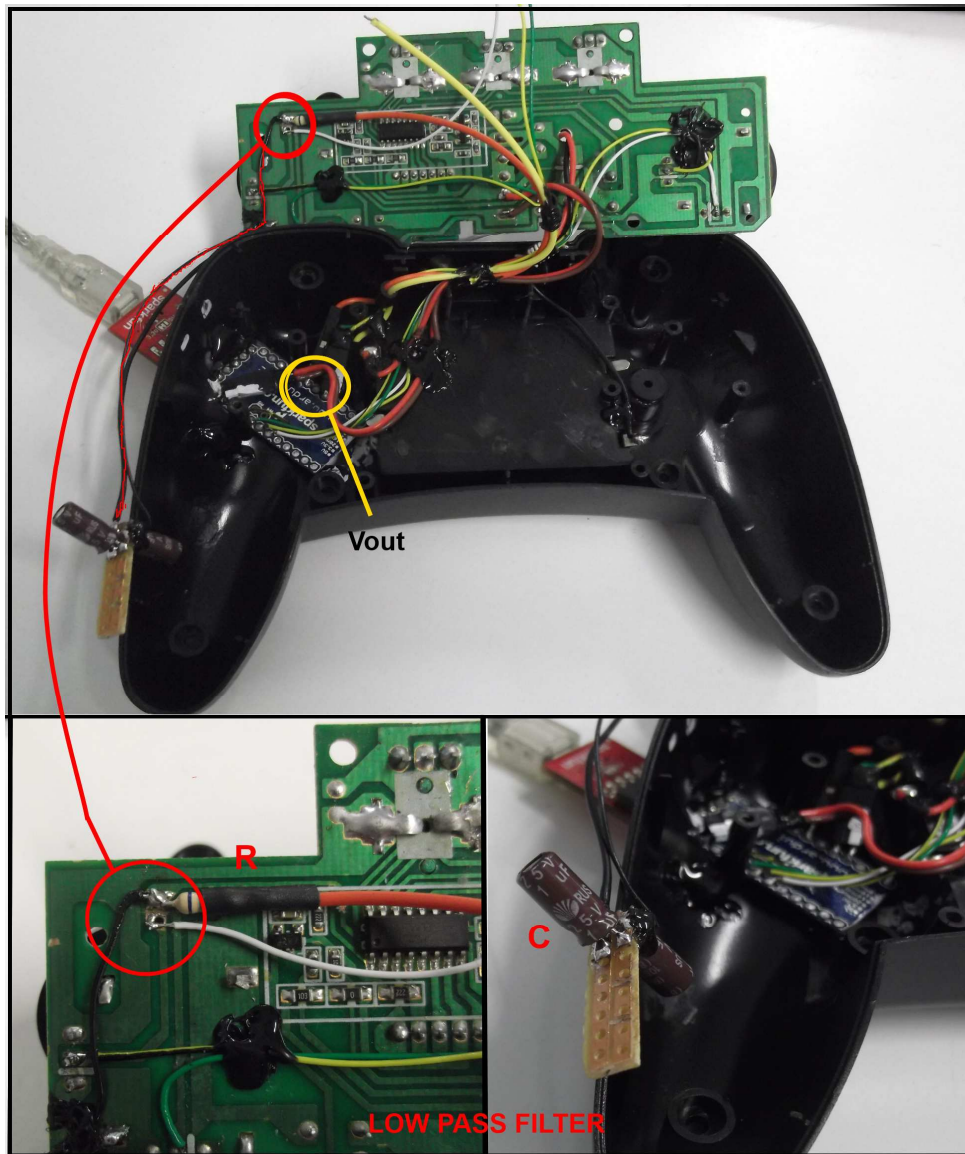
The system proposed and their characteristics (Figure 33):

$$\left. \begin{array}{l} C = 2.2\mu\text{F} \\ R = 7\text{k}\Omega \end{array} \right\} F_c = 10 \text{ Hz}$$



**Figure 33 – PWM filter desired response**

It will be necessary a filter for each output signal from Arduino (Tout, Yout, Pout and Rout), and then, wire the filter to the radio system input. Signal response is shown in Figure 31. Figure 34 describes PWM conditioning system implementation.



**Figure 34 – Arduino and PWM conditioning implementation**

The board should be modified in such a way the potentiometer is separated from radio processor Integrated Circuit (IC). The Arduino should read the potentiometer value, process the signal and then, through the PWM filter regenerate the analog signal. Figure 34 shows the details of this modification inside the manual control of the Quadrone.

#### **4.1.6 Intercom protocol**

The intercommunication systems are the antenna (radio system) which communicates with MAV platform and FTDI which connect the PC with Arduino microprocessor.

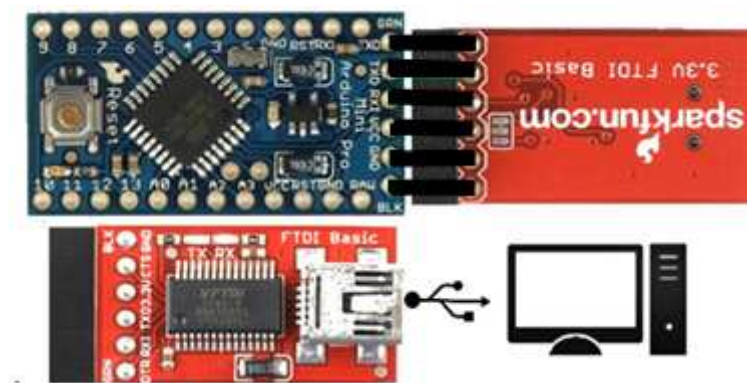


The first system, antenna, has been studied previously considered as a black box, where it is not suitable to actuate.

Second system, FTDI [106], converts UART from Arduino to USB in PC connection through a wire. It is designed to best interaction between PC (control unit) an Arduino processor (interpret).

FTDI converts single chip USB to asynchronous serial data transfer interface (UART). The FTDI also supply with 3,3V the Arduino when wired to USB port.

In this case, the communication will be byte a byte bidirectional (PC to Arduino and Arduino to PC). The implementation of FTDI is shown in Figure 35



**Figure 35 – FTDI connections (Red board)**

As, there can be seen, Arduino-FTDI connection has been made carefully in order to achieve a simple interaction when remote piloting case. There has been only necessary a small hole on back-lateral external case (Figure 36).



**Figure 36 – Arduino placement inside the manual control and FTDI to USB**

## 4.2 Software implementation solution

In this section will be described the software implementation both in microprocessor and PC application to perform the control system. The PC application will acts as a control unit, since PC acquires image data (through webcam), processes it to obtain a control command, and after, this command will be sent to the processor (Arduino) in remote control.

Arduino, in the other hand, has to process all the data received, from PC and remote-control commands (potentiometers), sends the data to PC and sends the data to radio system.

The two described processes are entire developed as a software application and programming.

### 4.2.1 Intercom protocol

There has been explained in section before how the communication takes part in terms of hardware implementation between PC and Arduino, through FTDI chip. Now, it is studied the data treat in communication process.

From Arduino characteristics in data communication:

- Speed communication set at 9600bps (default)
- Serial byte transmitting in binary code
- Bidirectional communication
- 4 different sources corresponding to MAV commands (Throttle, Yaw, Pitch, Roll)

There are many different ways to tackle the data transition, the solution proposed is a non-sequential organized frame. Every byte (8bits) corresponds to one command (Throttle, Yaw, Pitch, Roll) and represents its value. To associate the byte to corresponding command it will be necessary an identifier (ID). This ID (red haps in Figure 37) is the heading of the byte.

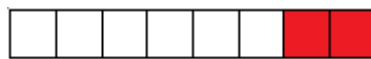


Figure 37 – Byte fragmentation

The Figure 37 shows the byte fragmentation proposed, with location reserved for byte ID. This ID has to be able to represent 4 different statuses (command).

Therefore, 2 bits are enough to become the byte ID, representing exactly the four different statuses. The rules implemented are shown in **¡Error! No se encuentra el origen de la referencia..**

Table 6 – Byte ID corresponds with command

Bits nomenclature	Integer	Command representation
00	0	Yaw
01	1	Throttle
10	2	Pitch
11	3	Roll



Consequently the resulting byte is a 6 bits location for data value and 2 bits location for ID. In Figure 37 white gaps represent bits for data value).

Now, it is important to understand what this method is representing for the system, because the system loses 2 bits in value information location. This process turns into a resolution disadvantage, since the system, now, is able to represent 64 values while, with entire byte for value information it could represent 256 different values.

Translating this 64 binary resolution to analog resolution, the result obtained is:

$$Resolution = \frac{range}{2^n - 1} = \frac{3,3}{63} = 0.05V \quad (4.2)$$

It is, through communication system, PC to Arduino, the processor only receives signal increments or decrements up to 0.05V, this value is less than MAV response resolution.

Concluding, the data treat is suitable for system response without affecting the resolution.

The advantages are that there is only needed a byte for each data that contains all the information. But, above all, there is not a determined a sequence of bytes, the byte is processed on time and not saved.

#### 4.2.2 Arduino programming

Remembering, the main Arduino functions are signal acquisition, data interpreting (processing), communication process and signal conditioning. This section exposes the most important parts of Arduino code which represents its.

The code is divided in three parts (Figure 38):

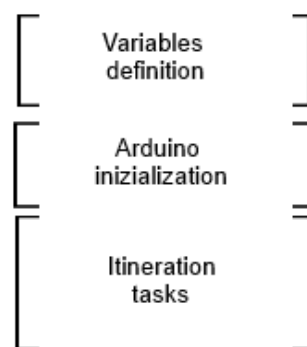


Figure 38 – Arduino code body

This sentence reads the input value from potentiometer, divides in 4, in order to adjust input signal to output signal in terms of resolution and finally assign to *mandoThrottle* variable.

```
mandoThrottle=analogRead(analogThrottle)/4;
```

These sentences employ a mask (Figure 39) to recognize the byte ID (bold sentences) which is located in last 2 bits. The mask eliminates (turn to 0) the data value of byte and maintain byte ID which is compared with throttle ID nomenclature.

```
if (Serial.available()) // Serial reading for throttle byte:
{
  byte a = Serial.read();
  else if ((a & B00000011) == B00000001){
    aByte[1] = a;
    pcThrottle = aByte[1] & B11111100;
    pcThrottleON = true; // Throttle data received from PC
  }
}
```



Figure 39 – Mask function

Then, the mask is inverted and the data value are processed, finally this values is assigned to *pcThrottle* variable and Boolean throttle variable becomes ON.

This sentences assign the final value to variable, it is, if the system has to obey the remote control orders or the PC orders. The decision depends if Boolean variable *pcThrottleON* is true or false.

```
// Subsystem assignment to final value
if(pcThrottleON)
  valThrottle=pcThrottle;
else
  valThrottle=mandoThrottle;
```

**If the system receives data from PC, the system obeys the PC commands.**

This sentence writes the value from *valThrottle* in the Arduino output PIN named *digThrottle* as a PWM signal.

```
analogWrite(digThrottle, valThrottle);
```

#### 4.2.3 Labview programming

Labview application allow the user designs systems with visual programming with a duality between block diagram and frontal panel (user interface). Image processing is one of the most significant tools supplied by Labview, allowing to do a complex image processing program easily.

There is needed a powerful tool that detects and processes a simple image pattern originating from the MAV platform and extract position data at real-time.

In the other hand, the data has to be processed and compared to make a control command. For these reasons Labview is chosen to design the control unit. The code programmed and user interface designed are in Annex 2.

The block diagram is divided in two main parts (two loops). There are two main iterations independent of each other.

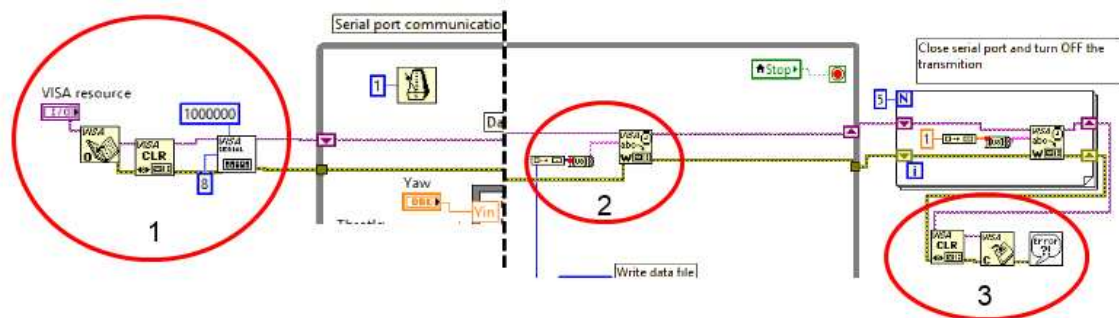
- First loop: is the faster loop for serial communication and data processing functions.
- Last loop: is a more complex loop for image processing and control unit functions.

The program replies to four important functions:

- Serial port communication (data transmission) function
- Virtual remote piloting function
- Image processing function
- Control unit function

### Serial port communication

This diagram (Figure 40) represents the functions relating to serial port COM.



**Figure 40 – Serial port communication Labview diagram (steps 1, 2 and 3)**

**Step 1:** Initialize Serial port with a byte communication, the port is chosen by the user in the frontal panel.

**Step 2:** The data (byte) is written in serial port and sent.

**Step 3:** When the loop finishes, the serial port is closed and cleaned, also the errors accumulated are indicated.

### Virtual remote piloting

Next diagram, Figure 41, shows the data setting for the different commands

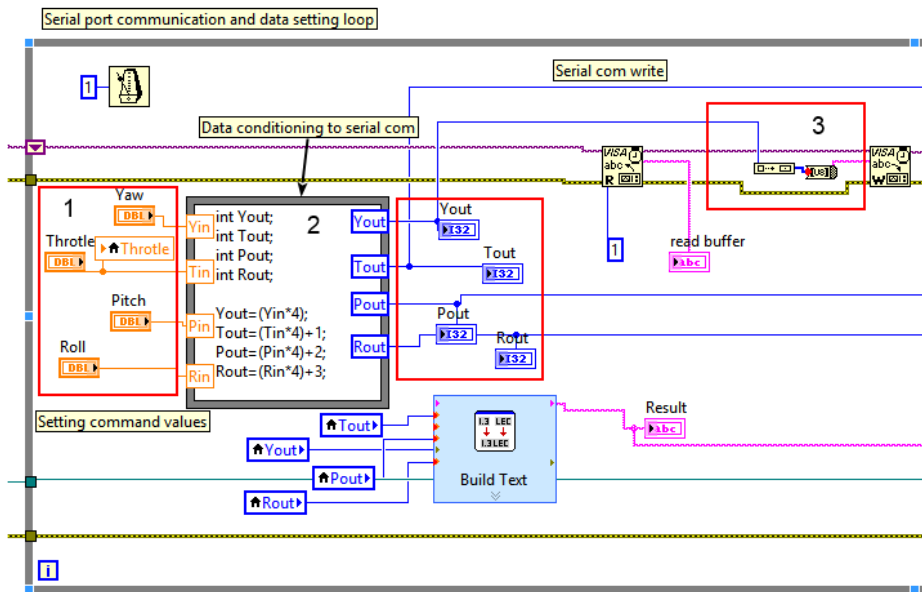


Figure 41 – Virtual remote piloting Labview diagram

**Step 1:** Values set by the user, an exception of Throttle value that can be changed automatically when autonomous navigation mode is ON.

**Step 2:** Value processing for data transmitting. The user sets the data simply ranged and it has to be translated into corresponding nomenclature, byte data (data value + ID). This process is explained in next section.

**Step 3:** Since, from step 2 is obtained an integer variable, to send the data as a byte it is necessary to convert the integer into string of bits.

### Image processing

Following diagram (Figure 42) advances the image processing process, despite of it is explained in next section.

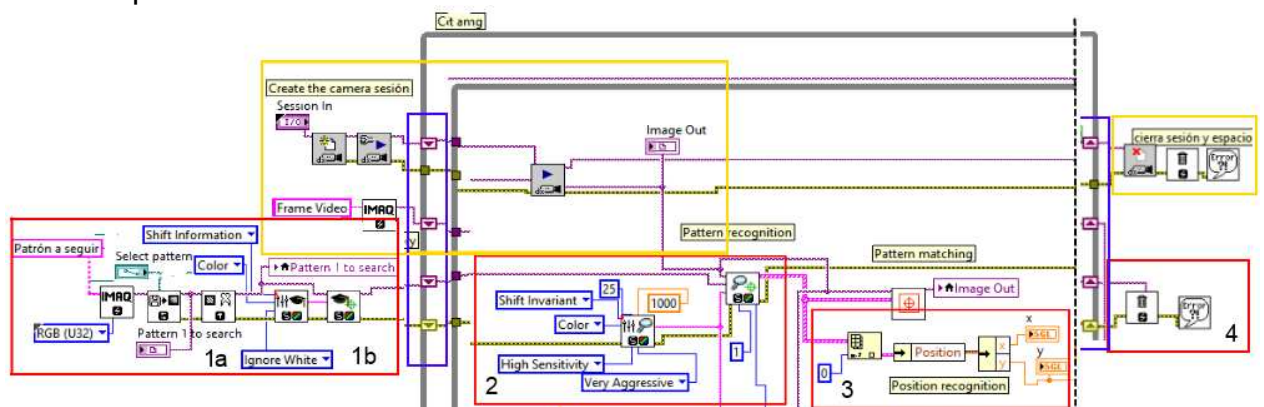


Figure 42 – Image processing Labview diagram

**Step 1:** Creates a temporary memory location for an image. Uploads the image pattern from user selected directory and extracts the image with adjustment of

resolution (1b). Then the program learns the pattern keeping the parameters specified.

**Step 2:** Searches the pattern received keeping the parameters specified in search setup. It is obtained an array of match clusters. It is only waited 1 match.

**Step 3:** Process match cluster and shows the first element corresponding to match position.

**Step 4:** Destroys images and frees the space they occupied in memory, also the errors occurred are indicated.

### Control unit

The final process done is the control process if indicated by user (Figure 43).

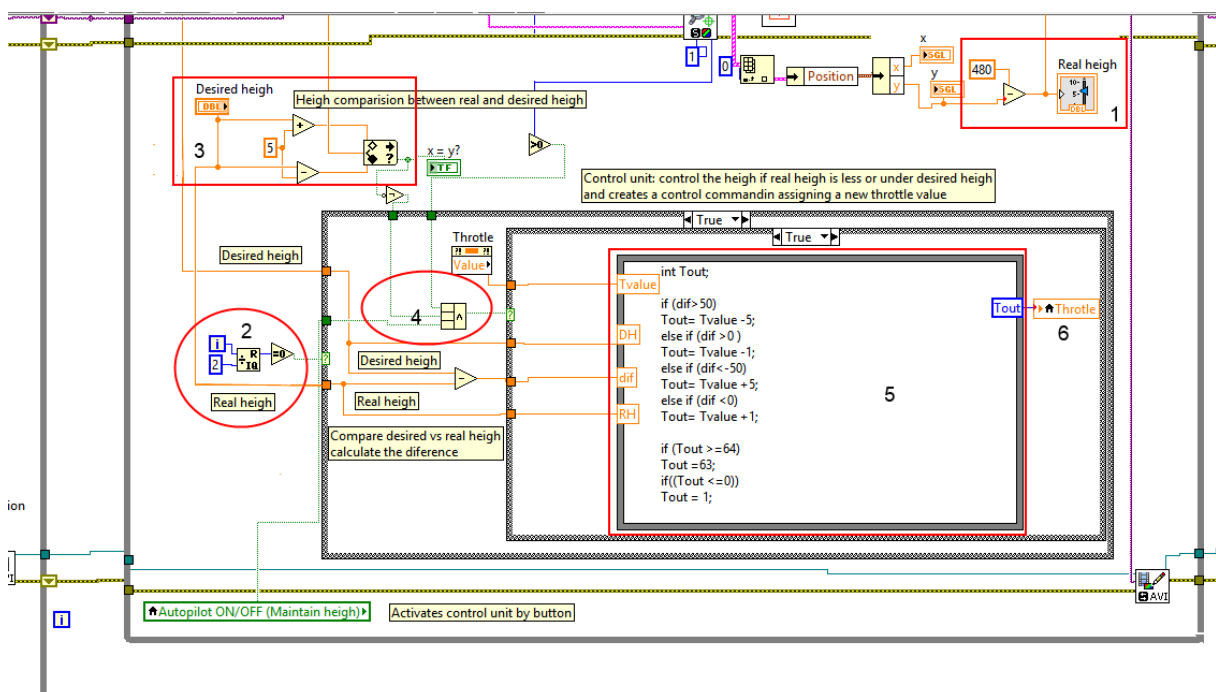


Figure 43 – Control unit Labview diagram

**Step 1:** Obtains the real height matched by image processing function and shows to the user (the height is in terms of resolution in pixels).

**Step 2:** Is a sub function to control the frequency of control unit function, since the MAV response is slower than function iteration. It is important to calibrate this function in order to adjust the system.

**Step 3:** Desired height is set by user and compared with real height. The comparing Boolean function have a resolution of  $\pm 5\text{px}$ , it is, if the difference between real height and desired height is  $\pm 5\text{px}$ , the function results FALSE. This resolution is enough for the MAV indoor application because the distance will be no more than 4 meters. In TRUE case, the control function could start (the outer case structure is true).

**Step 4:** If NOT comparison function from step 3, autonomous button and “match found” are TRUE, the system starts the control processing.

**Step 5:** Control process code to respond an output variable for Throttle.

**Step 6:** Throttle variable from control process is inserted in Throttle global variable.

#### 4.2.4 Remote piloting

Remote piloting mode allows the user control the MAV either by remote control or virtual control. Now, there is exposed how it works in the system (Figure 44).

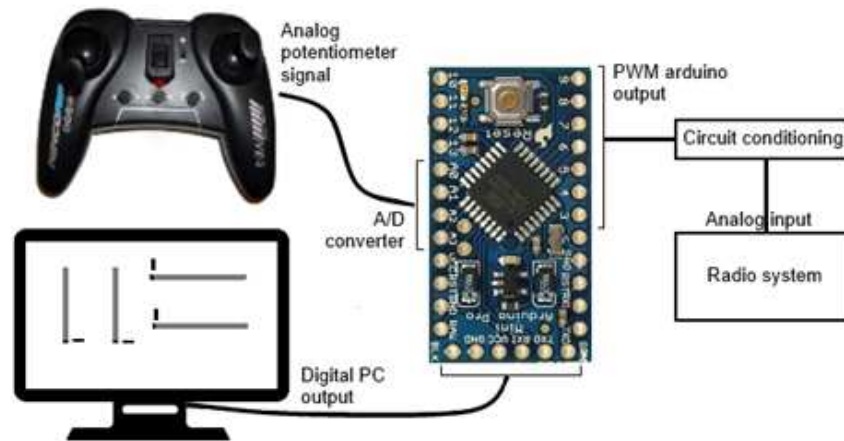



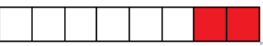
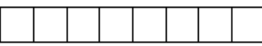



Figure 44 – Remote control mode function

It is taken into account that Arduino input signal (See also Table 7) has a 10 bits resolution and Arduino output signal has a 8 bits resolution, in terms of range value the relation is:

$$\frac{\text{input}}{\text{output}} = \frac{2^{10}}{2^8} = 4 \quad (4.2)$$

Table 7 – Original signal and result from Arduino processing

Signal origin	Arduino input	Arduino system	Arduino output
<b>Potentiometer signals</b> 0 to 3.3V (1 signal Vin from each command action) 	Convert to 10 bits resolution. $\text{inSignal} = \text{A/D} (\text{Vin})$ 	System resolution: $\text{Re } s = \frac{3,3V}{2^{10} - 1} = 0.003V$ Data processing (adapt to Arduino output): $\text{outSignal} = \text{inSignal}/4;$	PWM in 8 bits resolution (outSignal) 
<b>PC signals</b> Digital signal (Bin) (1 byte data) 6 bits data + 2 bits ID 	Decodification process (signal Bp): $\text{Bp} = \text{mask} (\text{Bin})$ 	System resolution: $\text{Re } s = \frac{3,3V}{2^6 - 1} = 0.05V$ Data processing: Value info OK: $\text{outSignal} = \text{Bp}$	PWM in 8 bits resolution (outSignal) 

If the Arduino receives any command from PC, it will obey this signal, otherwise Arduino will obey remote control commands.

On the other hand, remote control will be activated when PC is commanding and sets all the commands in 0.

#### ***4.2.5 Image recognition and processing***

To achieve a good control system, image acquisition and processing are very significant functions, which perform the sensing part of the control system (described in Chapter 3).

As well as, the camera hardware has an important role as an acquisition support; it is not studied in the project. It is focused in image recognition and processing.

There are analysed the different tools to tackle these functions in Labview, using the diagrams shown in last sections. The different tools are ordered described in the table below (Table 8).

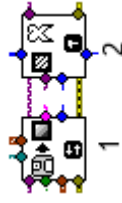
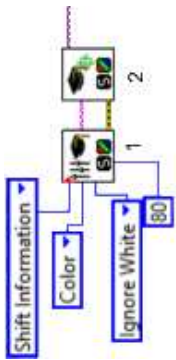
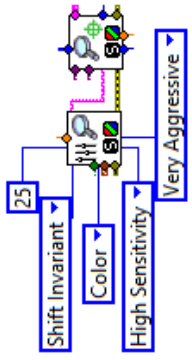

Function Name	Graphical representation	Description	Inputs needed	Outputs needed
Create a pattern to follow		1. Upload the image template to follow (from system directory). 2. Extracts the image with adjustment resolution.	- Image memory location. - File path directory - image in	- Image out
Learn colour pattern		1. Sets parameters that are used during the colour learning stage. 2. Creates a description of the colour template image	- Learn mode: set at shift information for invariant matching. - Feature mode: colour, since the LED pattern highlight by to colour properties. - Ignore white (chosen), white colour is not a distinctive value. - Saturation Threshold: not necessary	- Learn Colour Pattern Setup Data is a string that contains information about the setup parameters
Search colour pattern		1. Sets parameters that are used during the colour matching process. 2. Searches for a colour pattern in the input image.	- Minimum contrast: it is needed a high contrast since LED supply this distinctive. - Match mode: shift invariant (set in learn tool) - Colour sensitivity: The colour is a powerful property from pattern. - Search strategy: very aggressive, the colour in the template is uniform, the template is well contrasted from the background. - Currently image - Image template - Num. of matches' request: 1 match specified.	- Matches: is an array of match clusters, the first element is the position (point-coordinate cluster) where is obtained "y" position. Number of matches: is the number of template matches found in the inspection.
Draw Pattern match		Is a sub VI which draw, for each match, a bounding box and a landmark on the centre of the match.	- Matches: array of match cluster from search pattern tool to obtain position information. - image: Currently image	- Image: image resulting from source image and landmark.

Table 8 – Image processing functions





## CHAPTER 5. CHARACTERIZATION OF THE NEW AUTOPILOT MODE

This chapter collect the tests definition to check the system functionality in the initial system, remote control function, virtual remote and autonomous navigation function. Finally, there are exposed the results of these tests. The test

### 5.1 Flying tests

The tests realized to verify the system are strongly related to the time response and the signal characteristics.

And, above all, the MAV behaviour when autonomous system is activated, studying parameters as resolution, precision, response time, etc.

There are described the different flying test: the procedure and material. The results of developed system tests are described in next section.

#### 5.1.1 Initial system test

This test was developed before to start the improvements of system, with the aim to characterize the initial system. The results are important to good comparison with new solution.

The text can be seen in: <https://www.youtube.com/watch?v=KYFQGFxaOTw>

Next paragraphs are related with the development of the new control system.

#### 5.1.2 Remote control command test

This test (Figure 45) is performed with the aim to know the signal characterization and the MAV response in terms of propeller power.

There is compared the mechanical movement in throttle joystick (the corresponding signal value after potentiometer) with the Arduino A/D conversion and with MAV propellers response.

There will be seen the effects of the PWM conditioning and Arduino process in final response.

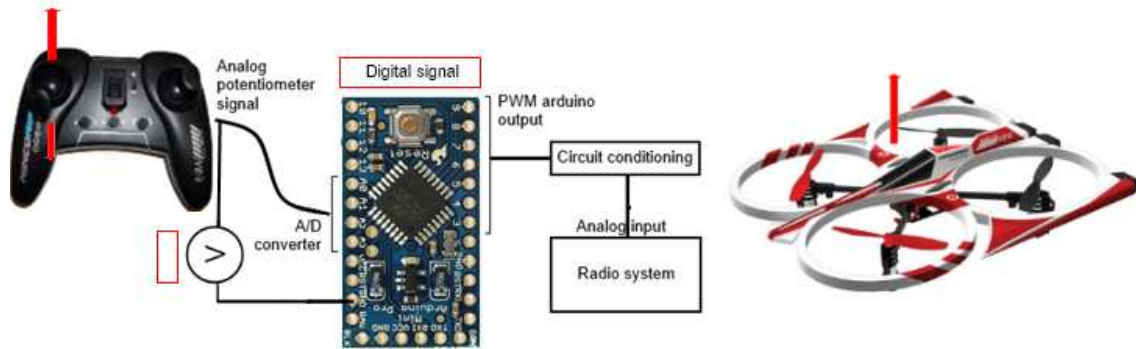


Figure 45 – Remote control command test procedure

### 5.1.3 Remote control virtual test

The goal of this test (Figure 46) is to know the digital signal characterization and the MAV response in terms of propeller power.

It is compared the virtually movement (the corresponding data byte) with the Arduino receiver and with MAV propellers response.

There will be seen the PWM conditioning, FTDI and Arduino processing effects in final response.

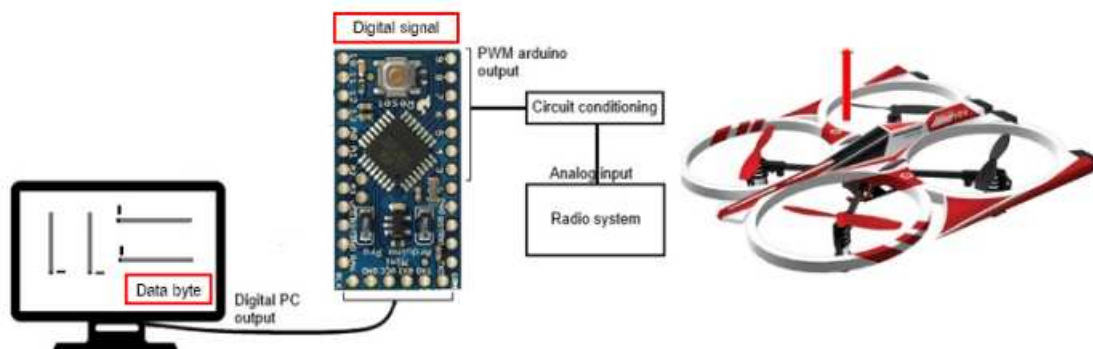


Figure 46 – Remote control virtual test procedure

### 5.1.4 Autonomous height control tests

The final test's aim is to check the autonomous control by image processing implemented. With the real time images in real time from Labview application and observing the MAV response. It will show image process and control process effects in final response and the **closed** loop efficacy. The study is qualitative.

## 5.2 Specifications of the system

There are described the results of the tests to extract system specifications.

### 5.2.1 Remote control command tests results

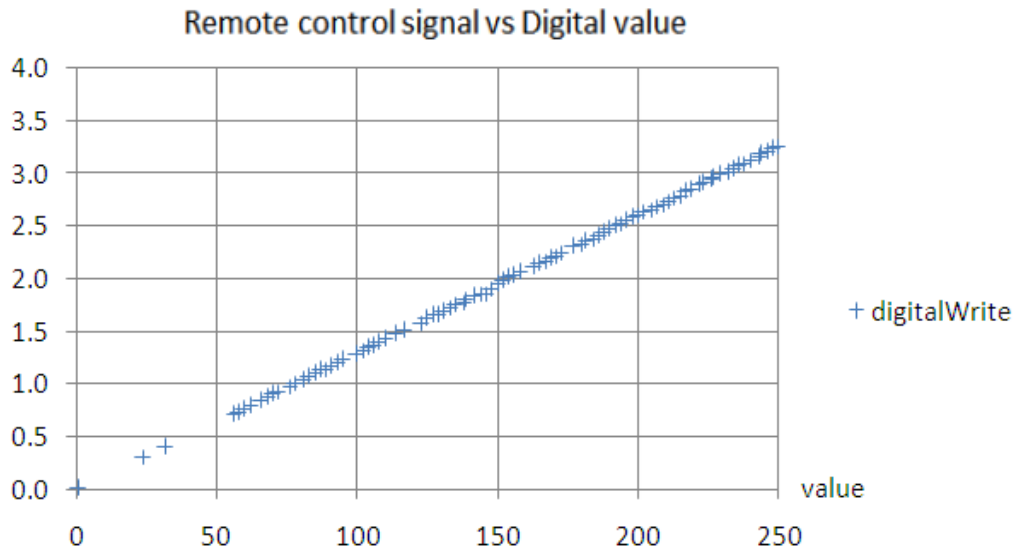


Figure 47 – Control potentiometer Voltage vs digital value by Arduino

As can be seen in the graphic, there are a linear response as it works in initial system with analog signal. The response is suitable for the system.

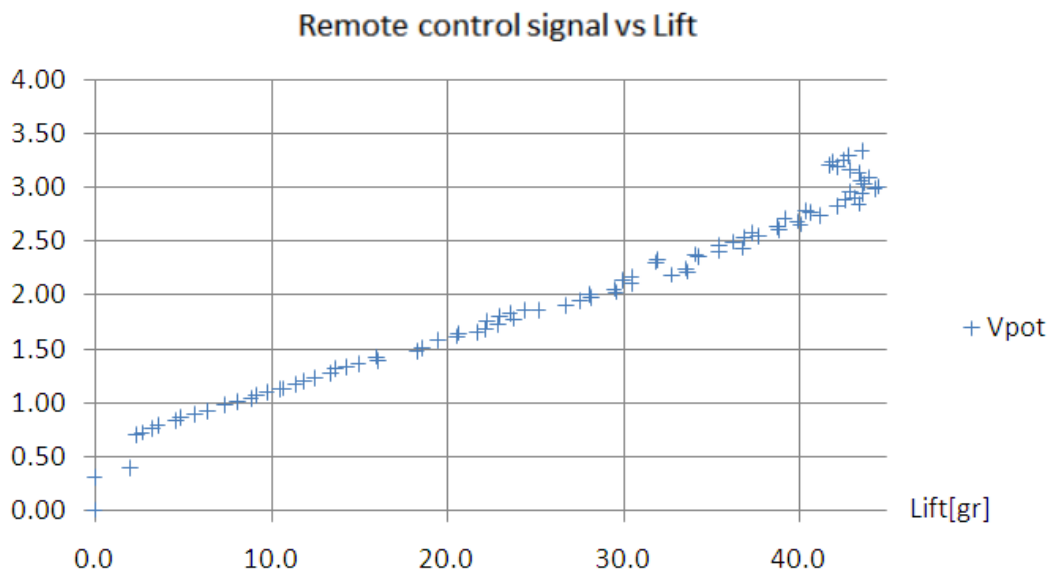


Figure 48 – Control potentiometer Voltage vs Total lift in grams

In this case, the response is lineal between 0.7 V and 3 V. Since the study is in terms of power, at flight beginning is needed a minimum power to raise the platform, this feature is reflexed between 0 V and 0.7 V, where exists high steps

between values. At the end, the power decreases due to continue demand to the battery and MAV instabilities that has to solve itself.

### 5.2.2 Remote control virtual tests results

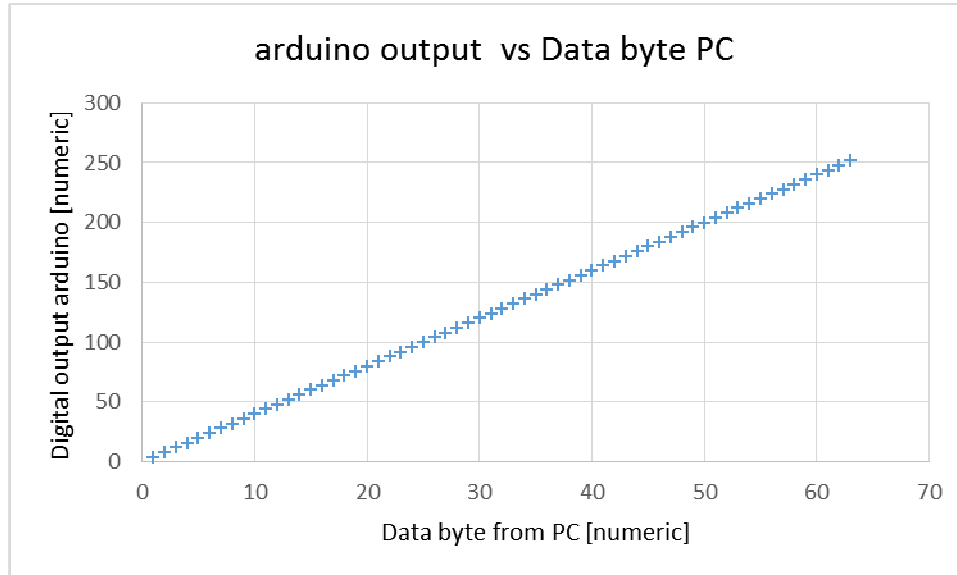


Figure 49 – Control virtually vs digital value in Arduino

The graphic shows a perfect linearity between the value set by the user and Arduino digital signal. The only problems that are contemplated: a bit loss or bit corrupted at communication moment (which is not happening in this test).

An error case is not disturbing, since the signal is sequential and transmission is fast enough.

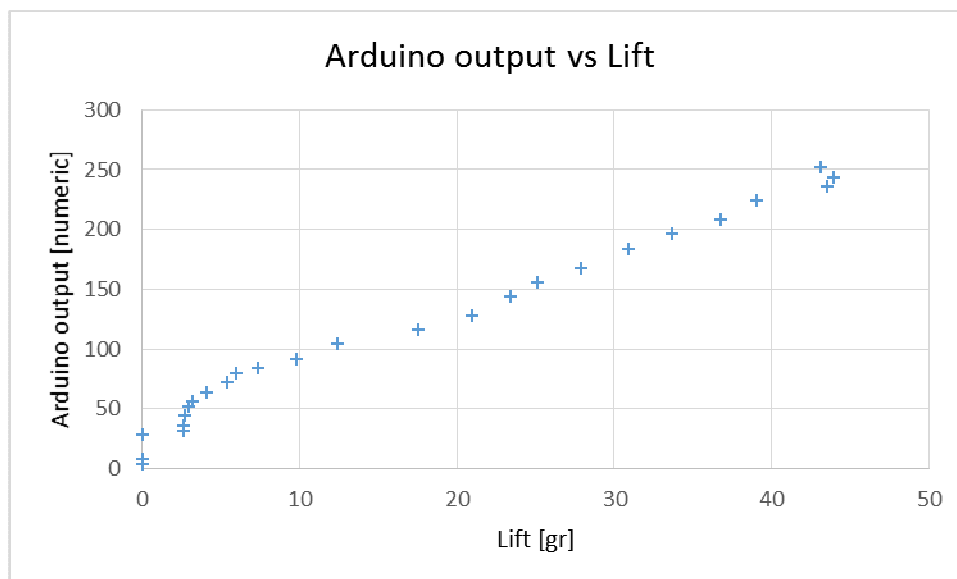
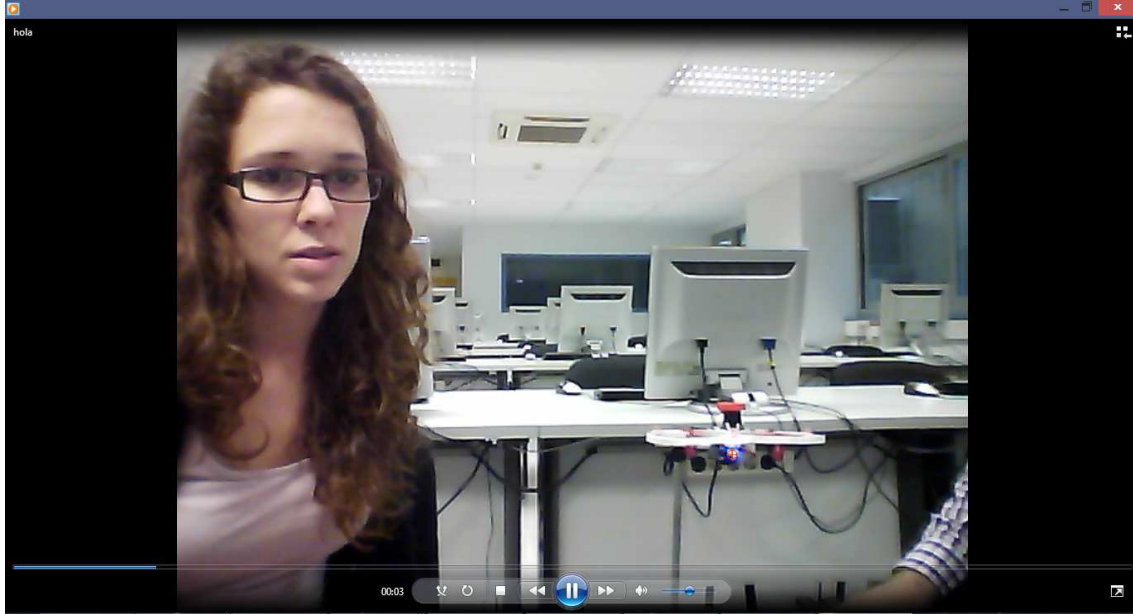


Figure 50 – Control virtually vs total lift in grams

This graphic is similar to remote control results (to the same test), although, the lift response for digital control is under the lift response for remote control, due to the small delay in data transmission.

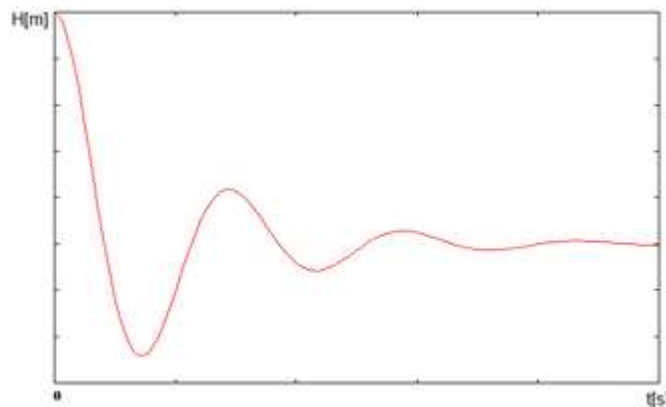
### 5.2.3 Remote control command test results



**Figure 51 – Real time image while control process**

The results are better appreciated in videos produced by Labview application when autonomous mode is ON. Figure 51 is an instant image from application when acquisition in real-time. The characteristics observed are:

- Image detection is very precise, with the exception of brightness white ambient.
- The MAV response is slower than control commands in some cases, however the time of control loop can be changed inside the application code. The result is an advance of control actuation from MAV final position.
- If target come out of camera scope, the control loop ends.
- With the autonomous system the dynamic stability becomes impaired, since with under damped response (Figure 52).



**Figure 52 – Under-damped response example**

### 5.3 Results analysis

Summarizing, the two different remote control systems respond faster and linearity as the initial system; the improvements made from initial system do not affect in MAV performances and response, moreover it has been increased the navigation accuracy, as the user know exactly the power employed.

In addition, image processing allows the user to know exactly the platform height (at the moment) and monitoring it.

When autonomous mode is ON, the system responds slower and the height control is not so accuracy as a remote mode.

## CHAPTER 6. CONCLUSIONS

This Chapter is composed by the conclusions, the future work and the environmental impact study.

### 6.1 General conclusions

This final Bachelor work has aimed to study, implement and validate a new control system for commercial MAV platform. A control system to navigate autonomously at least in z axis, this is commonly known as an autopilot that maintain height platform.

As it has been explained in the study, repeatedly, a Micro Air vehicle entails a strongly weight restrictions, which becomes a payload restriction.

For this reason, it is studied, compared and implemented an on ground control system that carries out the autonomous navigation.

Since, there has not been found similar solutions or similar projects, and there are an origin platform, the main work methodology has been reverse engineering.

It has been needed to study characterize and test the initial system, and after, check the proposed solutions. There are many tests and proposals that has not been included since there are as much as for other project.

The work has concluded satisfactory, the main goals has been achieved:

- The remote-control through Arduino both virtually and manual
- Image processing and tracking the platform
- Autonomous height control

The system is tested and developed for Quadrone initial platform, although it has not been developed for flaying-wing MAV there are described the basis to its implementation, therefore the only system to update would be the autonomous height control.

In general, the initial purposes was upgrade the realized work, since the secondary goals has not been accomplished. Due to the no response of the ICARUS group to implement this studies in their system, it was not possible to contact.

### 6.2 Work conclusions

Image processing system is able to detect the pattern (LED) and track it, but sometimes the program loses the LED when it is in white brightness places, and also when the template image is not enough accurate. This function could be improved in the future with a high level of image processing knowledge.



Moreover, the autonomous control function, which depends on image processing results will improve with the best results of image processing.

### **6.3 Future work**

The first improvements has to be produced in the image processing system, improving the characteristics to best pattern learning and detection. Moreover, in this section could be obtained more information from patter detection. For example if there are used two cameras, it could be possible extract 3D position.

In the other hand, the control system can be improved obtaining the transfer function of the system and studying a proper control actions in order change/adapt the damping, the settling time and the overshoot.

Labview requires an expensive license and it is a closed platform. All this program can be moved to other platform, programmed in C++, C#, to get an attractive visual interface, faster response and compatibility.

Finally, adapting the system to flying-wing MAV platform.

### **6.4 Environmental impact**

The MAV platform is an electrical object so the environmental impact is reduced.

Electronic components have been chosen carefully for a good system efficiency.

But, the impact of the MAV system must be appreciated by the environmental impact improvement that results from the applications and implementations of this system. For example, an indoor temperature control to best energy efficiency, a gas sensing, to detect gas leak, etc.





## BIBLIOGRAPHY

### References

- [1] Katsuhiko, Ogata., **Modern Control Engineering**, Pearson, fifth Edition, 1998. ISBN: 978-0-13-615673-4  
<http://books.google.es/books?isbn=8420536784>
- [2] Zdzislaw, Bubnicki, **Modern Control Theory**, Springer, First edition, 2005. ISBN: 978-3-540-23951-2
- [3] Goyal, Siddharth, **Study of a Longitudinal Autopilot For Different Aircrafts**, Punjab Engineering College (DU) Chandigarh, INDIA
- [4] Perez-Batlle, M., **Design and implementation of an autopilot for UAS**, UPC, 2012, UPCommons  
<http://hdl.handle.net/2099.1/16744>
- [5] Nelson, Robert C., **Flight Stability and Automatic Control**, McGra-Hill, 1989, ISBN: 0-07-046218-6
- [6] Blakelock, John H., **Automatic Control of Aircraft and Missiles**, Air Force Institute of Technology, John Wiley & Sons Inc, 2nd Edition, ISBN: 0-471-50651-6

### Web references

- [101] <http://www.thefreedictionary.com/control> (February 2014)
- [102] [http://en.wikipedia.org/wiki/Control\\_theory](http://en.wikipedia.org/wiki/Control_theory) (February 2014)
- [103] [http://en.wikipedia.org/wiki/Control\\_engineering](http://en.wikipedia.org/wiki/Control_engineering) (February 2014)
- [104] <https://www.youtube.com/watch?v=KYFQGFxaOTw> (July 2014)
- [105] <http://www.ni.com/labview/esa/>
- [106] <http://www.ftdichip.com/>



# ANNEXES

**TÍTOL DEL TFC:** Millores d'un petit vehicle aeri d'ala-volant tancant el llaç de control a terra

**TITULACIÓ:** Enginyeria Tècnica Aeronàutica, especialitat Aeronavegació

**AUTOR:** Irene Gallego Sánchez

**DIRECTOR:** Joshua Tristancho Martínez

**DATA:** 30 de Juny de 2014



## ANNEX A. Components datasheets

### A.1 Arduino Pro Mini Datasheet



#### Overview

The Arduino Pro Mini has 14 digital input/output pins (of which 6 can be used as PWM outputs), 8 analog inputs, an on-board resonator, a reset button, and holes for mounting pin headers. A six pin header can be connected to an FTDI cable or *Sparkfun* breakout board to provide USB power and communication to the board.

The Arduino Pro Mini is intended for semi-permanent installation in objects or exhibitions. The board comes without pre-mounted headers, allowing the use of various types of connectors or direct soldering of wires. The pin layout is compatible with the Arduino Mini.

There are two version of the Pro Mini. One runs at 3.3V and 8 MHz, the other at 5V and 16 MHz.

#### Summary

Microcontroller	ATmega168
Operating Voltage	3.3V or 5V (depending on model)
Input Voltage	3.35 -12 V (3.3V model) or 5 - 12 V (5V model)
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (of which 2 KB used by bootloader)
SRAM	1 KB
EEPROM	512 bytes
Clock Speed	8 MHz (3.3V model) or 16 MHz (5V model)

#### Power

The Arduino Pro Mini can be powered with an FTDI cable or breakout board connected to its six pin header, or with a regulated 3.3V or 5V supply (depending on the model) on the Vcc pin. There is a voltage regulator on board so it can accept voltage up to 12VDC. If you're supplying unregulated power to the board, be sure to connect to the "RAW" pin on not VCC.

The power pins are as follows:

- RAW: For supplying a raw voltage to the board.
- VCC: The regulated 3.3 or 5 volt supply.
- GND: Ground pins.

#### Input and Output

Each of the 14 digital pins on the Pro Mini can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 3.3. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:



- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the TX-0 and RX-1 pins of the six pin header.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the `analogWrite()` function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Pro Mini has 8 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). Four of them are on the headers on the edge of the board; two (inputs 4 and 5) on holes in the interior of the board. The analog inputs measure from ground to VCC. Additionally, some pins have specialized functionality:

- I2C: A4 (SDA) and A5 (SCL). Support I2C (TWI) communication using the Wire library.

There is another pin on the board:

- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## Communication

The Arduino Pro Mini has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega168 provides UART TTL serial communication, which is available on digital pins 0 (RX) and 1 (TX). The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board via a USB connection.

The ATmega168 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. To use the SPI communication, please see the ATmega168 datasheet.

## A.1 FTDI Basic Breakout 3.3V Datasheet



### Description

This is a basic breakout board for the FTDI FT232RL USB to serial IC. The pinout of this board matches the FTDI cable to work with official Arduino and cloned 3.3V Arduino boards. It can also be used for general serial applications. The major difference with this board is that it brings out the DTR pin as opposed to the RTS pin of the FTDI cable. The DTR pin allows an Arduino target to auto-reset when a new Sketch is downloaded. This is a really nice feature to have and allows a sketch to be downloaded without having to hit the reset button. This board will auto reset any Arduino board that has the reset pin brought out to a 6-pin connector.

The pins labelled BLK and GRN correspond to the colour wires on the FTDI cable. The black wire on the FTDI cable is GND, green is DTR. Use these BLK and GRN pins to align the FTDI basic board with your Arduino target.

There are pros and cons to the FTDI Cable vs the FTDI Basic. This board has TX and RX LEDs that allow you to actually see serial traffic on the LEDs to verify if the board is working, but this board requires a Mini-B cable. The FTDI Cable is well protected against the elements, but is large and cannot be embedded into a project as easily. The FTDI Basic uses DTR to cause a hardware reset where the FTDI cable uses the RTS signal.

This board was designed to decrease the cost of Arduino development and increase ease of use (the auto-reset feature rocks!). Our Arduino Pro and LilyPad boards use this type of connector

## ANNEX B. Programming code

### B.1 Arduino Programming code

```
//Variable definition
int digThrottle = 3; // PWM Pin OUT Throttle
int digYaw = 5; // PWM Pin OUT Yaw
int digPitch = 6; // PWM Pin OUT Pitch
int digRoll = 9; // PWM Pin OUT Roll
int analogThrottle = 3; // Analog read PIN Throttle
int analogYaw = 2; // Analog read PIN Yaw
int analogPitch = 1; // Analog read PIN Pitch
int analogRoll = 0; // Analog read PIN Roll
int valThrottle = 0; // assignment value
int valYaw = 0; // assignment value
int valPitch = 0; // assignment value
int valRoll = 0; // assignment value
int pcYaw=0; // Input value PC Yaw
int pcThrottle=0; // Input value PC Throttle
int pcPitch=0; // Input value PC Pitch
int pcRoll=0; // Input value PC Roll
int mandoYaw =0;
int mandoThrottle =0;
int mandoPitch =0;
int mandoRoll =0;
int axisByte = 0;
int i=1024;
int aByte[4]={0,0,0,0}; // Array of bytes
boolean pcYawON = false;
boolean pcThrottleON = false;
boolean pcPitchON = false;
boolean pcRollON = false;

//Initialize Arduino PINS
void setup()
{
  pinMode(digThrottle, OUTPUT); //Initialize the digThrottle PIN as OUTPUT
  pinMode(digYaw, OUTPUT);
  pinMode(digPitch, OUTPUT);
  pinMode(digRoll, OUTPUT);
  Serial.begin(9600); //Initialize the serial COM in 9600bps
}

//Itineration tasks
void loop()
{
  // Read PC data
  if (Serial.available())
  {
    byte a = Serial.read();

    //Bytes masks, last two bits motion information
    if((a & B00000011) == B00000000) {
      aByte[0] = a;
      pcYaw = aByte[0] & B11111100;
      pcYawON = true; // Yaw data received from PC
    }
    else if((a & B00000011) == B00000001){
      aByte[1] = a;
      pcThrottle = aByte[1] & B11111100;
      pcThrottleON = true; // Throttle data received from PC
    }
    else if((a & B00000011) == B00000010){
      aByte[2] = a;
      pcPitch = aByte[2] & B11111100;
      pcPitchON = true; // Pitch data received from PC
    }
    else {
      aByte[3] = a;
      pcRoll = aByte[3] & B11111100;
      pcRollON = true; // Roll data received from PC
    }
  }

  if(aByte[0]==0 && aByte[1]==1 && aByte[2]==2 && aByte[3]==3)
  //If all the values are null disconnect from PC
  {
    Serial.println("not connection");
    pcYawON = false;
  }
}
```

```
    pcThrottleON = false;
    pcPitchON = false;
    pcRollON = false;
  }
  Serial.println(a,DEC);
}

// Remote control reading data
mandoThrottle=analogRead(analogThrottle)/4;
mandoYaw=analogRead(analogYaw)/4;
mandoPitch=analogRead(analogPitch)/4;
mandoRoll=analogRead(analogRoll)/4;

// Subsystem assignment to final value
if(pcThrottleON)
  valThrottle=pcThrottle;
else
  valThrottle=mandoThrottle;

valYaw=mandoYaw;
valPitch=mandoPitch;
valRoll=mandoRoll;
//Serial.println(valThrottle); //muestra el valor que recibe
Serial.write(valThrottle); // sends final value to PC

//writes the value in output PINS
analogwrite(digYaw, valYaw);
analogwrite(digThrottle, valThrottle);
analogwrite(digPitch, valPitch);
analogwrite(digRoll, valRoll);
}
```

## B.2 Labview code

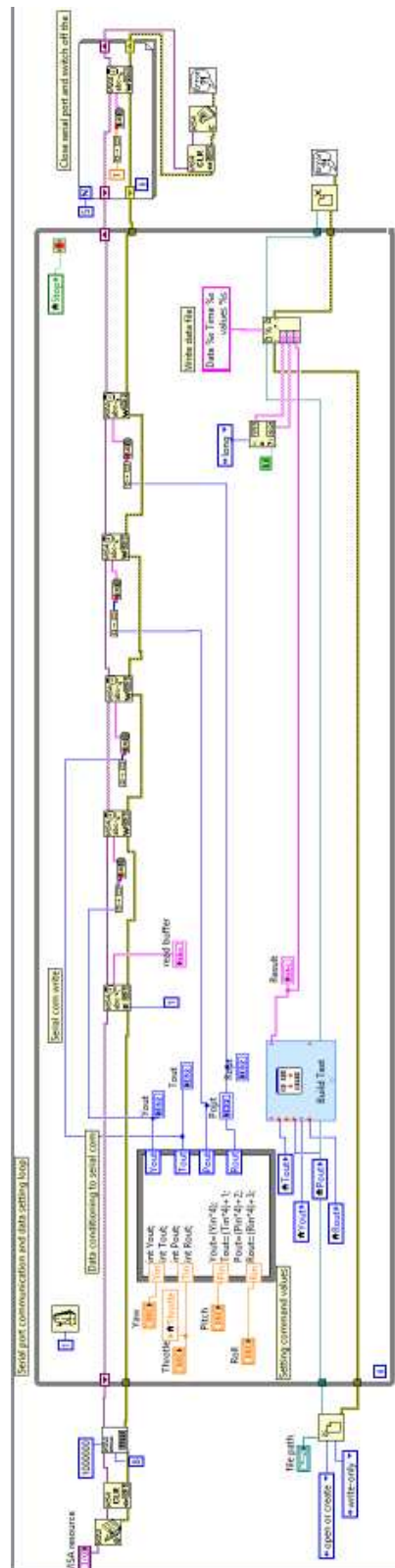


Figure 53 – Communication and command loop code diagram

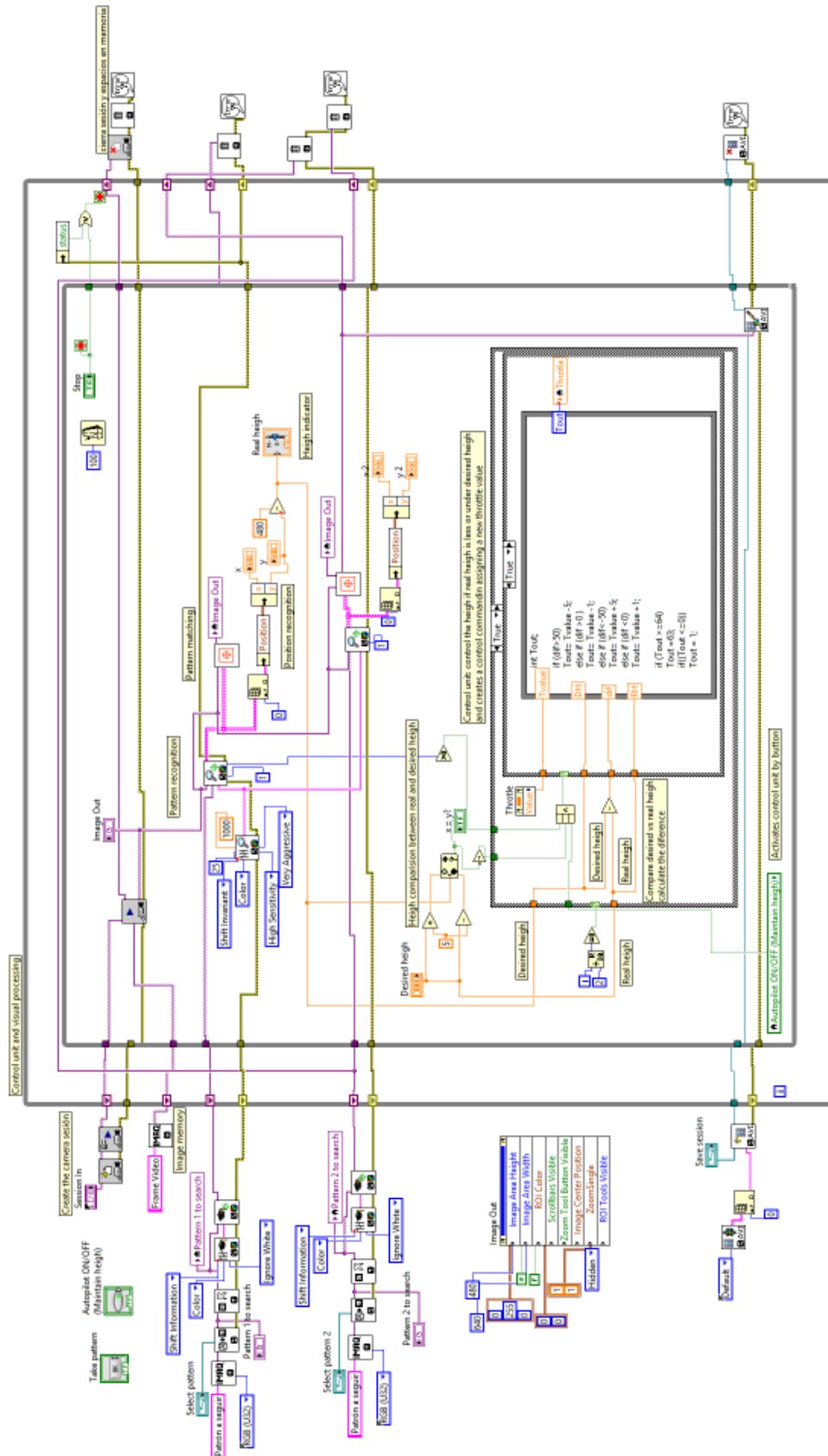


Figure 54 – Image processing and autonomous control loop code diagram

### B.3 Labview user interface

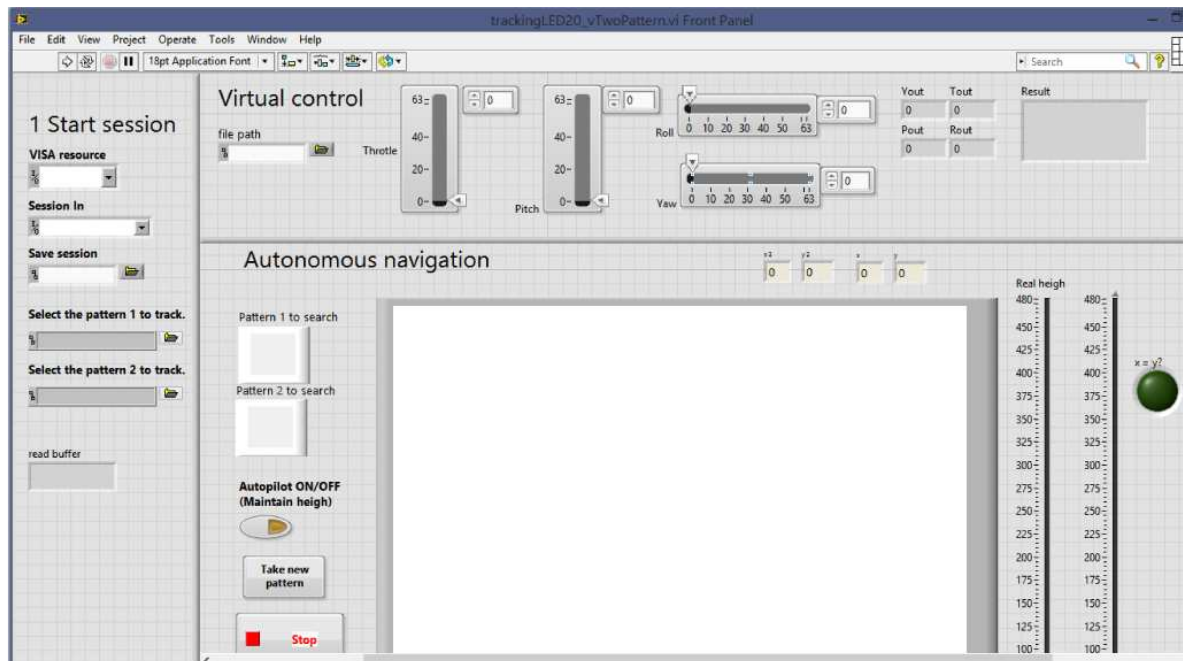


Figure 55 – Complete user interface view

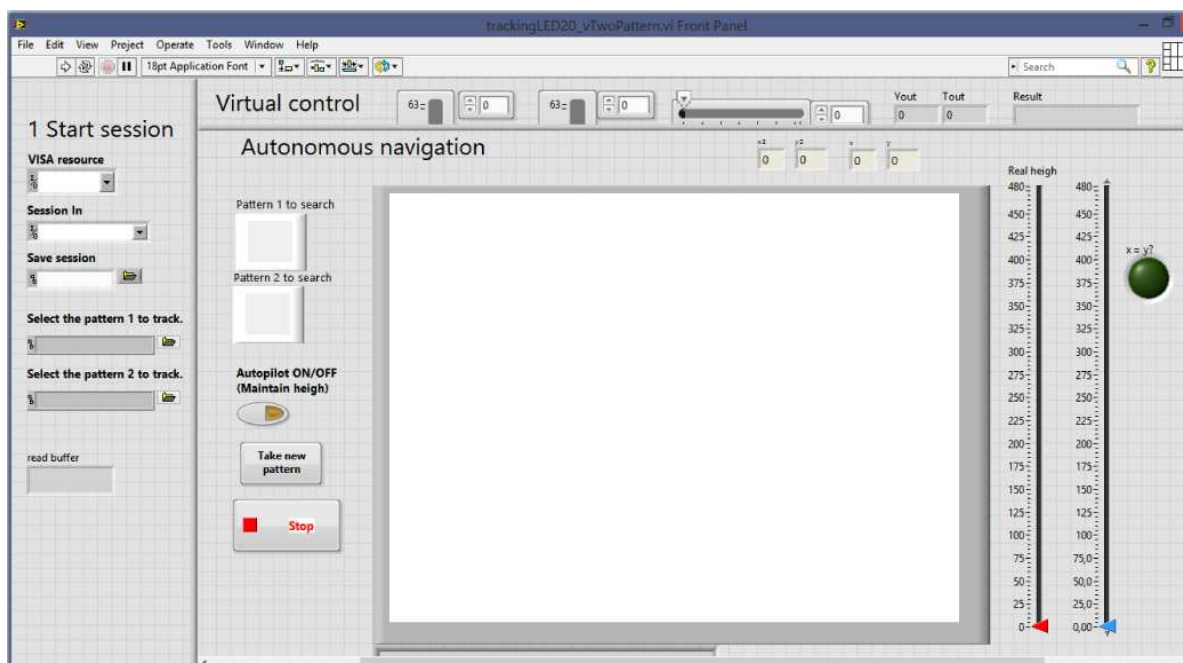


Figure 56 – Autonomous navigation detail user interface

