ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ & ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Ανάπτυξη και Ρύθμιση Αυτόματου Πιλότου Αεροπλάνου Μικρής Κλίμακας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Γεώργιου Ζωγόπουλου Παπαλιάκου

Επιβλέπων: Κώστας Κυριακόπουλος

Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2013

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ & ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Ανάπτυξη και Ρύθμιση Αυτόματου Πιλότου Αεροπλάνου Μικρής Κλίμακας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Γεώργιου Ζωγόπουλου Παπαλιάκου

Επιβλέπων: Κώστας Κυριακόπουλος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 25 Ιουλίου 2013.

.....	.....	.....
Κ. Κυριακόπουλος	Κ. Τζαφέστας	Κ. Πεκμεστζή
Καθηγητής Ε.Μ.Π.	Επ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2013

.....

Γεώργιος Ζωγόπουλος Παπαλιάκος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Τα μη επανδρωμένα εναέρια συστήματα (UAS) έχουν κερδίσει σημαντική θέση στο προσκήνιο των εξελίξεων της αεροναυπηγικής. Παραδοσιακά, όπως συνήθως, το υπόβαθρό τους ήταν στρατιωτικής προέλευσης, όμως την τελευταία δεκαετία πολυάριθμες εταιρίες, πανεπιστήμια και ομάδες ανά τον κόσμο αναπτύσσουν μη επανδρωμένα εναέρια συστήματα που ενσωματώνονται όλο και περισσότερο στις ανάγκες και την καθημερινότητα του σύγχρονου κόσμου. Παρ' όλ' αυτά, σε καμία περίπτωση δεν είναι ένα εξαντλημένο πεδίο ούτε κοινή γνώση. Οι νέες μεθοδολογίες, αλγόριθμοι, εφαρμογές και η ενσωμάτωσή τους σε ένα ενιαίο σύστημα είναι αντικείμενο έρευνας και η τεχνογνωσία που ακοκομίζεται είναι πολύτιμη.

Στο πνεύμα αυτό, το θέμα της διπλωματικής εργασίας αυτής είναι να μυήσει τον γράφοντα στην κατασκευή αυτόνομων μη επανδρωμένων εναέριων συστημάτων, εστιάζοντας σε ατράκτους με σταθερή πτέρυγα, λαμβάνοντας υπ' όψιν όλες τις παραμέτρους που μπορεί να συμβάλλουν στο τελικό αποτέλεσμα.

Λέξεις Κλειδιά

Μη Επανδρωμένο Αεροσκάφος, Μη Επανδρωμένο Εναέριο Σύστημα, Αυτόματος Πιλότος, Τηλεκατευθυνόμενο Αεροσκάφος, Ενσωματωμένο Σύστημα.

Abstract

Unmanned Aerial Systems (UASs) have gained a significant place in the aerospace industry. Traditionally, as expected, they had a military background, but in the last decade multiple companies, universities and groups all around the world develop UASs that respond to emerging social needs. However, this is neither a depleted field nor common knowledge. New methodologies, algorithms, applications and their integration into a combined system are the object of ongoing research and the knowhow gained in the process is valuable.

In that aspect, the goal of this diploma thesis is to initiate the author into the creation of UASs, focusing on fixed-wing airframes, taking into account all the factors that affect the final result.

Key Words

Unmanned Aerial Vehicle, Unmanned Aerial System, Autopilot, Radio-Controlled Aircraft, Embedded System.

Ευχαριστίες

Για την εκπόνηση της εργασίας αυτής συντέλεσαν ποικίλοι παράγοντες:

Ο σημαντικότερος συντελεστής και παράγοντας που διαμόρφωσε την μέχρι τώρα πορεία μου στη ζωή (άρα και την εκπόνηση αυτής της εργασίας) ήταν η οικογένειά μου και οι συγγενείς μου. Δεν θα μπορούσα να ζητήσω περισσότερη υποστήριξη, κατανόηση και παρότρυνση από αυτή που μου προσέφεραν. Σας ευχαριστώ πολύ όλους.

Ειδικά θέλω να ευχαριστήσω τους γονείς μου και τον Νίκο και την Ελένη. Χάρη σ' αυτούς ξέρω πια ποια θέλω να είναι η θέση μου στον κόσμο αυτόν.

Ακόμη, ευχαριστώ τον καθηγητή μου, Κώστα Κυριακόπουλο, για την εμπιστοσύνη που μου έδειξε και μου δείχνει, ακόμα κι αν εγώ ακόμη δεν είμαι σίγουρος για τον εαυτό μου μερικές φορές.

Σημαντική ήταν και η προσφορά της ευρύτερης διαδικτυακής κοινότητας των φόρουμ www.allaboutcircuits.com και www.diydrones.com, γιατί μου έδειξαν ότι η τεχνολογία και το επιστημονικό πνεύμα δεν είναι ούτε προνόμιο λίγων, ούτε κλείνεται σε τέσσερις τοίχους.

Τέλος, θα ήθελα να ευχαριστήσω τους θαμώνες του άτυπου μοντελοδρόμιου του αεροδρομίου του Ελληνικού, που αγκάλιασαν την έρευνά μου και διευκόλυναν σημαντικά την αποπεράτωσή της.

Acknowledgment

Many factors contributed to the making of this diploma thesis:

The most important influence on my course in life so far (thus the development of this thesis as well) was my family and relatives. I couldn't ask for more support, understanding and encouragement than I received. I thank you all deeply.

Especially, I am grateful to my parents and Nick and Helen. Thanks to them, I know by now who I want to be in this world.

Furthermore, I want to thank my professor, Kostas Kyriakopoulos, for the trust he showed (and still does) to me, even if I'm not sure about myself sometimes.

Also, the internet communities of the fora www.allaboutcircuits.com and www.diydrones.com, played an important role, since they showed me that technology and the scientific spirit are not a privilege but a common good and can't be constrained within the walls of classroom.

Finally, I want to thank all the resident pilots of the unofficial model airfield of Elliniko, for embracing my efforts and helping me throughout the duration of the thesis.

Nomenclature

b	Wingspan
$\overline{c}$	Mean aerodynamic chord
C_{X_Y}	Coefficient of magnitude X, in regard to the magnitude Y
g	Acceleration of gravity
$\mathbf{J}$	Inertial matrix
l	Torque on body x-axis
m	Torque on body y-axis **OR** total mass of airframe
n	Torque on body z-axis
p	Angular velocity on x-axis
q	Angular velocity on y-axis
r	Angular velocity on z-axis
u	Velocity on body x-axis
v	Velocity on body y-axis
V_a	Airspeed
w	Velocity on body z-axis
α	Angle of attack
β	Sideslip angle
γ	Path angle
δ_a	Aileron control variable
δ_e	Elevator control variable
δ_r	Rudder control variable
δ_t	Throttle control variable

Table of Contents

Table of Figures

Table of Tables

1 Introduction

1.1 Verbal problem statement

The goal of this thesis statement is to describe the development process of an autopilot system for small scale radio controlled fixed-wing aircraft. The autopilot should be able to take over the guidance of the aircraft and complete the flight mission prerequisites.

1.2 Social Need

This is a response to the increasing trend of introducing robotics to all facets of daily life and providing safe, cheap and user-friendly solutions to emerging needs.

Piloting small scale Radio Controlled aircraft has seen a great rise in popularity in the last decade. Advances in foam, battery and motor technology have made the hobby accessible and affordable. Many novices have taken up the hobby and the now larger user base is pushing it to new bounds. People are finding new ways to take advantage of that technological platform; First Person View (FPV) flight is one example, which allows the pilot to feel as if he performs free-flight in real time, in a highly interactive experience. Other users are trying to integrate another favourite field into RC modeling: robotics.

At the same time, robotics has also been rendered more accessible for similar reasons as well as the downsizing of sensors and processing units. The popularity of smartphones and tablets has enabled microelectronics to do great leaps and produce small, light and cheap hardware. Control systems amateurs have formed a new market for semiconductor companies, who now compete over who will provide the most interesting and easy to use products.

The combination of the above is applied robotics in RC aircraft, a trend that is rapidly picking up and provides the consumer with new ways to harness the educational potential and the “cool factor” of robots.

1.3 Technical Problem Statement

The goal of this thesis project was to design and build a standalone autopilot module for use in model RC fixed wing aircraft. The module should be mounted inside the aircraft and operate autonomously, while allowing for user intervention during error conditions. The solution will be based on an onboard microcontroller, equipped with the necessary sensors, which will form an autonomous guidance unit responsible for

completing a flight mission. It should require the minimum amount of modifications on the airframe in order to operate, in other words, be a non-intrusive installation.

The module should include the following features:

1. Take over the control surfaces of the aircraft and perform attitude control
2. Navigate the aircraft through a set of predefined waypoints
3. Perform take-off autonomously off of a flat, hard surface, such as concrete.
4. Keep a flight log in a removable storage medium.
5. Interface with a computer program in order to unload the flight log data for processing.

Emerging problems are:

a. The implementation of a real-time orientation estimation solution, necessary for any control effort.
b. Giving the robot spatial awareness in order to fulfill the flight mission.
c. Develop an adequate inner loop control regime as well as a navigation controller.
d. Building a physical system capable of carrying out the module objectives while keeping a low budget.

1.4 Parallel work

In the last few years, autopilot systems for toy RC aircraft have been developed by companies, open source communities or a combination of both.

Notable examples are:

- Ardupilot by 3DRobotics
- OpenPilot
- PX4, an ETH spinoff project
- Paparazzi
- DJI products

1.5 Thesis Structure

This document will follow the following structure:

In Chapter 2, definitions and general information is given on the subject of aeronautics and RC modelling.

In Chapter 3, an analysis/review is made on the kinematics and dynamics of a fixed-wing aircraft.

In Chapter 4, the solution process of the problem is expanded upon.

In Chapter 5, issues that came up during the research process are discussed.

In Chapter 6, results and performance of the proposed solution are presented.

2 Definitions

2.1 Reference Frames

As in all problems in robotics, various quantities can appear in different frames of coordinates and there is the need for transformation of those quantities between frames.

Earth frame - F^i

Also known as the inertial frame, $\{x^i, y^i, z^i\}$. This orthocanonical set of coordinates describes the location of objects of interest relatively to an unmoving observer on the earth. The origin of this frame is arbitrary but usually it is placed on the surface of the earth, at the launch point of the aircraft.

The X-Y plane of the frame is tangent to the surface of the Earth at the location of the origin. The X axis points towards the North. Usually, the geographical North is selected, instead of the magnetic one. The Z-axis points straight down towards the earth, aligned with the force of gravity. The Y-axis is defined by the other two axes and points towards the East.

Body frame - F^b

This orthocanonical set of coordinates, $\{x^b, y^b, z^b\}$, measures quantities as seen from onboard the aircraft. Motors and sensors of the aircraft operate on this frame. The origin of this frame is located on the center of gravity of the aircraft.

The x-axis points forward, along the longitudinal axis of the aircraft. The y-axis points to the right, towards the wing of the aircraft. The z-axis is defined by the other two axes and points downwards.

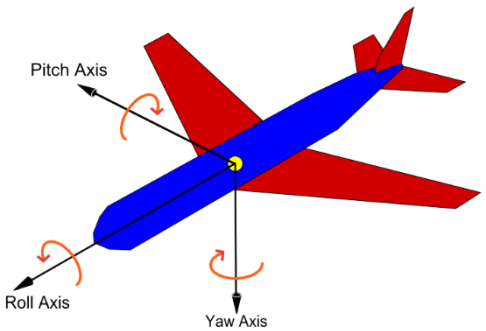


Figure 1: Aircraft Body Axes

Stability frame - F^s

The air mass situated above the area of operation of the aircraft can generally be moving relatively to the ground. This is commonly described and felt as wind.

However, this has implications in the aerodynamics of the aircraft; for example, airplanes produce lift that is related to the speed they move at, in relation to the air mass that surrounds them, known as relative wind, not related to the ground speed. Thus, there is a need to distinguish this speed with ground speed. In general, the relative wind affects the aerodynamics of the aircraft, whereas the ground speed is taken into account in the motion planning of the aircraft.

The relative wind vector V is defined as the motion of the air mass around the aircraft and can generally have components in all x^b, y^b and z^b directions.

We need to define a new coordinate frame in order to describe the aerodynamic effects on the airframe. A first step is to define the stability frame $\{x^s, y^s, z^s\}$. The origin of the frame is located on the aerodynamic center of the aircraft. Usually, by design choice, this coincides with the center of mass. y^s is aligned with y^b and x^s is in the same plane as the relative wind vector and y^s.

Wind frame – F^w

The second step is to define the wind frame (x^w, y^w, z^w). z^w is aligned with z^s and x^w is aligned with the wind vector.

Airspeed V_a is defined as the magnitude of the wind vector.

2.2 Euler Angles

The Euler angles make up a set of three angles that define the transformation from the Earth frame to the body frame. Many definitions exist for this set, but in aviation the Tait-Bryan convention is mostly used. This is comprised by the set of angles $\{\phi, \theta, \psi\}$ defined as follows:

The ψ angle is the angle of rotation around z^i that aligns x^i with the projection of x^b on the x^i-y^i plane. We will call the resulting frame F^1.

θ is the angle of rotation around y^1 that aligns x^1 with x^b. We will call the resulting frame F^2.

ϕ is the angle of rotation around x^2 (which is the same as x^b) that aligns y^2 with y^b.

ϕ, θ and ψ are also often referred to as roll, pitch and yaw respectively. Roll and yaw are also known as bank and heading.

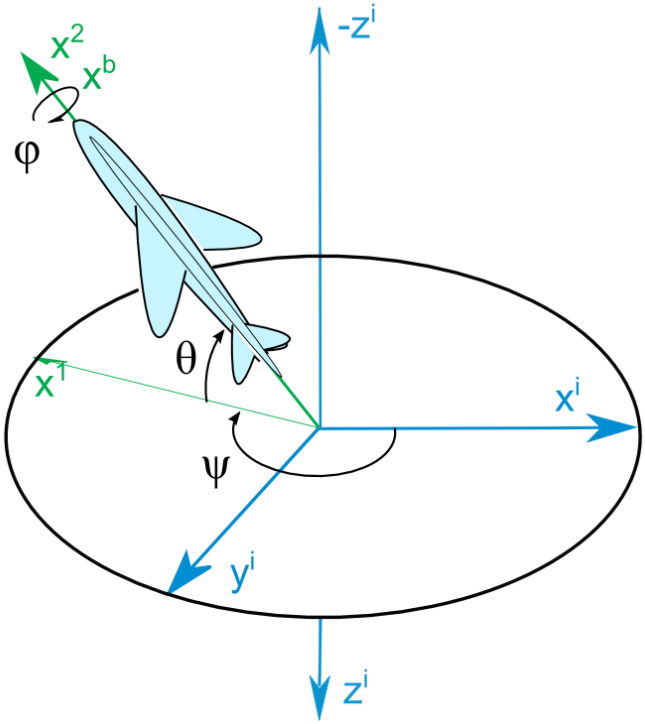


Figure 2: Euler Angles

2.3 Angle of attack and sideslip angle

Two equivalently important angles are the rotation transformation angles that link the body frame and the wind frame.

The **negative** (left-handed) rotation angle about y^b that aligns x^b with the projection of the wind vector on the (x^b, x^b) plane (or equivalently with x^s) is called angle of attack and is denoted by α. This is an important magnitude, as airplanes need to maintain a positive angle of attack in order to produce sufficient lift. The inverted angle orientation is used for conventional reasons.

The rotation angle about z^s that aligns the x^s with the wind vector (or equivalently with x^w), is called sideslip angle and is denoted by β. This expresses the amount of sidewind that is experienced by the airframe.

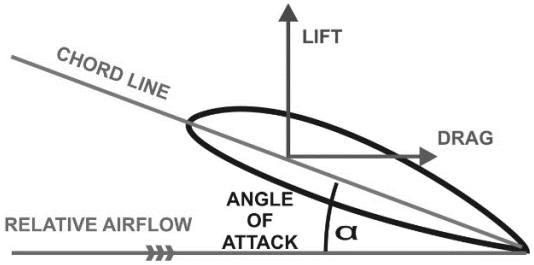


Figure 3: Angle of Attack

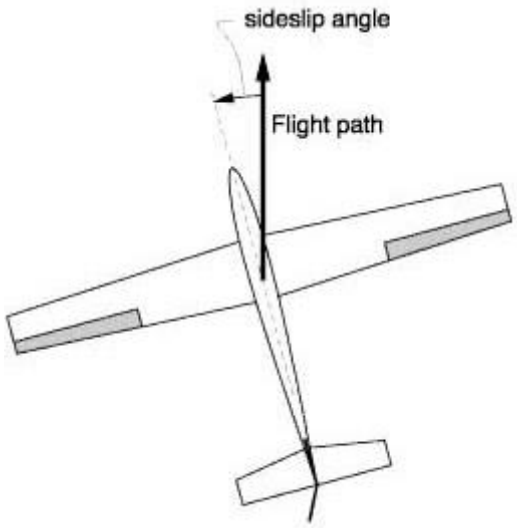


Figure 4: Sideslip Angle

2.4 Aircraft position

We define as p_n and p_e the components of the projection of the position vector of the airframe, measured in F^i. p_n is the component along the North direction, p_e is the component along the East direction.

2.5 Course and Flight-path angle

Since we have taken into account sideward wind during the flight of the airframe, we should also be aware of its effect during navigation. Even though the airframe might point straight towards a certain direction, wind might be pushing it sideward. As a result, the overall travel direction can be different from the heading angle. This direction, expressed in F^i, is called course and is symbolized by the Greek letter χ.

The flight-path angle is defined as $\gamma = \theta - \alpha$ and is the angle between x^i and the projection of the velocity vector of the airframe onto the (x^i, z^i) plane, measured in F^i. It expresses the overall angle at which the aircraft climbs or descends, with the angle of attack taken into account.

2.6 Altitude

Throughout this thesis altitude is denoted by h and is measured along z^i. That means that when the airframe is above the ground, it has negative altitude. This is done in accordance to how the inertial frame is defined, and in practice doesn't cause any significant complication.

We all have an existing perception of what altitude is. However, in aviation, altitude has a dual nature.

One way to define altitude is by the geometric quantity measured by the GPS receiver that is situated onboard an aircraft. This is relative to the ellipsoid that models the shape of the Earth and is used by the GPS. The most commonly used ellipsoid model is the WGS-84.

Another measurement method is through the barometric pressure. In that way, altitude can be calculated as a function of barometric pressure and temperature and be extrapolated, while following the International Standard Atmosphere (ISA) model. It should be noted, though, that this altitude measurement is knowingly using only an approximation model and thus a constant altitude measurement between two different days can vary as a geometrical quantity.

For conventional and other reasons, the barometric altitude is primarily used in aviation (Graham, 2011).

2.7 Lateral/Longitudinal Quantities

While analyzing the aircraft system and planning its control, we will make a grouping of the quantities involved, judging by the planes of the F^b they present themselves at. This can be also seen as a qualitative distinction.

We will include in the longitudinal quantities of the aircraft the altitude, airspeed, pitch angle, elevator input and propeller thrust. These are considered to present themselves in the (x^b, z^b) plane.

We will include in the lateral quantities of the aircraft the yaw angle, roll angle and aileron input. These are considered to present themselves in the (x^b, y^b) plane.

2.8 Introduction to RC airplanes

In the interest of providing a background on the operation of Radio Controlled fixed wing aircraft to the reader, a short description of the common setups will be given.

The airframe follows more or less the existing concepts of aeronautics. It can be constructed from wood, metal, acrylic glass, carbon fibres or more modernly and commonly polypropylene or polyolefin foam. Its weight ranges from a few hundred grams to a few kilograms.

The thrust is produced from a propeller motor situated in the nose (tractor configuration) or the back/top (pusher configuration) of the fuselage. The motor can be either electric or internal combustion type. Electric motors are powered by multi-

cell LiPo batteries. Large IC motors may use common gasoline and smaller ones typically use nitromethane.

The control surfaces are the ailerons (on the main wing), the elevators (on the horizontal stabilizer) and the rudder (on the vertical stabilizer). Flaps can also be found on RC models. The control surfaces are motorized by servomotors powered by an onboard battery.

The motor thrust and servomotor angles are controlled by PWM signals produced by the radio controlled receiver, placed on board. The receiver is powered by an onboard battery.

The receiver is linked to the transmitter/controller, held and operated on the ground by the pilot. The radio frequencies used for the link are situated in the open/hobby frequencies of the radio spectrum and vary from country to country. Most common frequencies are 27MHz, 35MHz, 433MHz, 900MHz, 1.2GHz, 2.4GHz and 5.8GHz. The 2.4GHz band is especially common in modern transmitters/receivers.

3 Kinematics and Dynamics Revision

Since the controllers used by the autopilot system operate according to a dynamic and kinematic model of the airframe, and certain assumptions about it, in this section this model will be described. This process follows the work of R. Beard and T. McLain, which also spans extensively onto other chapters of this thesis (Beard & McLain, 2012).

3.1 Kinematics

The relation that links the rates of the Euler angles with the angular rates in the body frame $\boldsymbol{\omega}_{b/i}^{b} = [p\ q\ r]^T$ is

$$\begin{aligned}\begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R_2^b(\phi)\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_2^b R_1^2 \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix}\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}\end{aligned} \tag{3.1}$$

Inverting the rotation matrix we get

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}\begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.2}$$

Similarly, we express the derivative of the airframe position in the inertial frame in regard to the airframe velocity $\mathbf{v}^b = [u\ v\ w]^T$, expressed in the body frame:

$$\begin{aligned}\frac{d}{dt}\begin{bmatrix} p_n \\ p_e \\ -h \end{bmatrix} &= R_b^i \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ &= \left(R_i^b\right)^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ &= \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}\begin{bmatrix} u \\ v \\ w \end{bmatrix}\end{aligned} \tag{3.3}$$

3.2 Dynamics

We define $\mathbf{f}^b = (f_x, f_y, f_z)^T$ as the force vector applied on the airframe, measured in F^b. From the equation of Coriolis we get

$$m\left(\frac{d\mathbf{v}^b}{dt} + \boldsymbol{\omega}_{b/i}^b \times \mathbf{v}^b\right) = \mathbf{f}^b \Leftrightarrow$$
$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m}\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{3.4}$$

Next, we introduce the inertia matrix $\mathbf{J}$ of the airframe

$$\mathbf{J} = \begin{bmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{bmatrix}$$

However, usually in fixed wing aircraft, there usually is geometrical symmetry around the (x^b, z^b) plane. This results in zero elements in the matrix.

$$\mathbf{J} = \begin{bmatrix} J_x & 0 & -J_{xz} \\ 0 & J_y & 0 \\ -J_{xz} & 0 & J_z \end{bmatrix} \tag{3.5}$$

The inverse of the inertia matrix is

$$\mathbf{J}^{-1} = \begin{bmatrix} \frac{J_z}{\Gamma} & 0 & \frac{J_{xz}}{\Gamma} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{\Gamma} & 0 & \frac{J_x}{\Gamma} \end{bmatrix} \tag{3.6}$$

Where

$$\Gamma = J_x J_z - J_{xz}^2 \tag{3.7}$$

The moment vector expressed in F^b is $\mathbf{m}^b = [l\ m\ n]^T$ and the angular momentum vector is $\mathbf{h}^b = \mathbf{J}\boldsymbol{\omega}^b$.

We apply the Coriolis equation once more, this time on $\mathbf{m}^b$ and $\mathbf{h}^b$:

$$\frac{d\mathbf{h}^b}{dt} + \boldsymbol{\omega}_{b/i}^b \times \mathbf{h}^b = \mathbf{m}^b \tag{3.8}$$

Solving for the derivative of the angular rates we get

$$\dot{\boldsymbol{\omega}}_{b/i}^{b} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \mathbf{J}^{-1}\left[-\boldsymbol{\omega}_{b/i}^{b} \times (\mathbf{J}\boldsymbol{\omega}_{b/i}^{b}) + \mathbf{m}^{b} \right] \tag{3.9}$$

Expanding, we get

$$\begin{aligned} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} &= \begin{bmatrix} \frac{J_z}{\Gamma} & 0 & \frac{J_{xz}}{\Gamma} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{\Gamma} & 0 & \frac{J_x}{\Gamma} \end{bmatrix} \left(\begin{bmatrix} J_{xz}pq + (J_y - J_z)qr \\ J_{xz}(r^2 - p^2) + (J_z - J_x)pr \\ (J_x - J_y)pq - J_{xz}qr \end{bmatrix} + \begin{bmatrix} l \\ m \\ n \end{bmatrix} \right) \\ &= \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + \frac{1}{J_y} m \\ \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \end{bmatrix} \end{aligned} \tag{3.10}$$

Where

$$\begin{aligned} \Gamma_1 &= \frac{J_{xz}(J_x - J_y + J_z)}{\Gamma} \\ \Gamma_2 &= \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma} \\ \Gamma_3 &= \frac{J_z}{\Gamma} \\ \Gamma_4 &= \frac{J_{xz}}{\Gamma} \\ \Gamma_5 &= \frac{J_z - J_x}{J_y} \\ \Gamma_6 &= \frac{J_{xz}}{\Gamma} \\ \Gamma_7 &= \frac{(J_x - J_y)J_x + J_{xz}^2}{\Gamma} \\ \Gamma_8 &= \frac{J_x}{\Gamma} \end{aligned} \tag{3.11}$$

Next, we proceed to define the forces and moments exerted on the airframe.

Gravity

Gravity exerts only forces, since, by definition, originates in the center of gravity.

$$\mathbf{f}_g = \begin{bmatrix} f_{gx} \\ f_{gy} \\ f_{gz} \end{bmatrix} = mg \begin{bmatrix} \sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{bmatrix} \tag{3.12}$$

$\mathbf{f}_g$ is expressed in the body frame.

Aerodynamics

Aerodynamics play a significant role in the airframe model. Keep in mind that aerodynamic forces and moments are defined in the stability frame.

The forces and moments exerted in the x^s, y^s and z^s axes are named $\{Drag,\ f_y,\ Lift\}$ and $\{l, m, n\}$ respectively.

They all depend on the mean dynamic pressure that acts upon the airframe, which is dependent on the airspeed and the air density.

$$\bar{q} = \frac{1}{2}\rho V_\alpha^2 \tag{3.13}$$

Generally, aerodynamic forces and moments are complicated functions of many variables:

$$\begin{aligned} Drag &= \bar{q}SC_D(\alpha, q, \delta_e) \\ f_y &= \bar{q}SC_Y(\beta, p, r, \delta_a, \delta_r) \\ Lift &= \bar{q}SC_L(\alpha, q, \delta_e) \\ l &= \bar{q}SbC_l(\beta, p, r, \delta_a, \delta_r) \\ m &= \bar{q}S\bar{c}C_m(\alpha, q, \delta_e) \\ n &= \bar{q}SbC_n(\beta, p, r, \delta_a, \delta_r) \end{aligned} \tag{3.14}$$

However, it is useful to linearize those functions around the state of level flight. This linearization is valid while uniform flow is maintained below and above the wings and for most non-aggressive maneuvers. The resulting functions are

$$
\begin{aligned}
Drag &= \bar{q}S\left(C_{D_0} + C_{D_\alpha}\alpha + C_{D_q}\frac{\bar{c}}{2V_a}q + C_{D_{\delta_e}}\delta_e\right)\\
f_y &= \bar{q}S\left(C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right)\\
Lift &= \bar{q}S\left(C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\frac{\bar{c}}{2V_a}q + C_{L_{\delta_e}}\delta_e\right)\\
l &= \bar{q}Sb\left(C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{b}{2V_a}r + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r\right)\\
m &= \bar{q}Sc\left(C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}}{2V_a}q + C_{m_{\delta_e}}\delta_e\right)\\
n &= \bar{q}Sb\left(C_{n_0} + C_{n_\beta}\beta + C_{n_p}\frac{b}{2V_a}p + C_{n_r}\frac{b}{2V_a}r + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r\right)
\end{aligned}
\tag{3.15}
$$

Finally, we need to transform the set of equations onto the body frame:

$$
\begin{aligned}
f_x &= \bar{q}S\left(C_{X_0} + C_{X_\alpha}\alpha + C_{X_q}\frac{\bar{c}}{2V_a}q + C_{X_{\delta_e}}\delta_e\right)\\
f_y &= \bar{q}S\left(C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r\right)\\
f_z &= \bar{q}S\left(C_{Z_0} + C_{Z_\alpha}\alpha + C_{Z_q}\frac{\bar{c}}{2V_a}q + C_{Z_{\delta_e}}\delta_e\right)\\
l &= \bar{q}Sb\left(C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{b}{2V_a}r + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r\right)\\
m &= \bar{q}Sc\left(C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\bar{c}}{2V_a}q + C_{m_{\delta_e}}\delta_e\right)\\
n &= \bar{q}Sb\left(C_{n_0} + C_{n_\beta}\beta + C_{n_p}\frac{b}{2V_a}p + C_{n_r}\frac{b}{2V_a}r + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r\right)
\end{aligned}
\tag{3.16}
$$

Where

$$
\begin{aligned}
C_{X_0} &= -C_{D_0}\cos\alpha + C_{L_0}\sin\alpha \\
C_{X_\alpha} &= -C_{D_\alpha}\cos\alpha + C_{L_\alpha}\sin\alpha \\
C_{X_q} &= -C_{D_q}\cos\alpha + C_{L_q}\sin\alpha \\
C_{X_{\delta_e}} &= -C_{D_{\delta_e}}\cos\alpha + C_{L_{\delta_e}}\sin\alpha \\
C_{Z_0} &= -C_{D_0}\sin\alpha - C_{L_0}\cos\alpha \\
C_{Z_\alpha} &= -C_{D_\alpha}\sin\alpha - C_{L_\alpha}\cos\alpha \\
C_{Z_q} &= -C_{D_q}\sin\alpha - C_{L_q}\cos\alpha \\
C_{Z_{\delta_e}} &= -C_{D_{\delta_e}}\sin\alpha - C_{L_{\delta_e}}\cos\alpha
\end{aligned}
\tag{3.17}
$$

Finally, we examine the forces and moments produced by the propulsion, expressed in the body frame. In the airframe used in the thesis, and in many cases, the motor is placed in the nose of the fuselage and its axis is aligned with x^b.

In most electric motors, where an Electronic Speed Controller (ESC) is in charge of RPM regulation, the exit speed of the airstream coming out of the propeller is

$$V_{exit} = k\delta_t \tag{3.18}$$

where k is a coefficient linking the two quantities and is generally constant.

The upstream and downstream pressure of the airstream are

$$
\begin{aligned}
P_{upstream} &= P_0 + \frac{1}{2}\rho V_a^2 \\
P_{downstream} &= P_0 + \frac{1}{2}\rho V_{exit}^2
\end{aligned}
\tag{3.19}
$$

Where P_0 is the ambient atmospheric pressure.

The force generated by the propeller is

$$
\begin{aligned}
f_{xp} &= S_{prop}(P_{downstream} - P_{upstream}) \\
&= \frac{1}{2}\rho S_{prop}\left((k\delta_t)^2 - V_a^2\right)
\end{aligned}
\tag{3.20}
$$

The torque produced by the motor along x^b is

$$m_{xp} = -k_{T_p}(k_\Omega\delta_t)^2 \tag{3.21}$$

Where $k_\Omega\delta_t$ is the angular velocity of the propeller and k_{T_p} is the constant coefficient linking it with the torque produced.

To sum up, the total forces and torques applied on the airframe are

$$\begin{aligned}\Sigma\mathbf{f} &= \mathbf{f}_{gravity} + \mathbf{f}_{aerodynamic} + \mathbf{f}_{propeller} \\ \Sigma\mathbf{m} &= \mathbf{m}_{aerodynamic} + \mathbf{m}_{propeller}\end{aligned} \tag{3.22}$$

Defining $\mathbf{x}$ as a generalized state vector including (a, b, p, q, r) and $\boldsymbol{\delta}$ as the input vector containing $(\delta_a, \delta_e, \delta_r, \delta_t)$, and using (3.16) on (3.10), we can incorporate the dynamics and the control inputs to the kinematics:

$$\begin{aligned}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} &= \begin{bmatrix}\Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6(p^2 - r^2) \\ \Gamma_7 pq - \Gamma_1 qr\end{bmatrix} + \bar{q}S\begin{bmatrix} b(\Gamma_3 C_l(\mathbf{x},\boldsymbol{\delta}) + \Gamma_4 C_n(\mathbf{x},\boldsymbol{\delta})) \\ \frac{\bar{c}}{J_y} C_m(\mathbf{x},\boldsymbol{\delta}) \\ b(\Gamma_4(C_l(\mathbf{x},\boldsymbol{\delta}) + \Gamma_8 C_n(\mathbf{x},\boldsymbol{\delta}))\end{bmatrix} \\ &= \begin{bmatrix}\Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6(p^2 - r^2) \\ \Gamma_7 pq - \Gamma_1 qr\end{bmatrix} + \bar{q}S\begin{bmatrix} bC_p(\mathbf{x},\boldsymbol{\delta}) \\ \frac{\bar{c}}{J_y} C_m(\mathbf{x},\boldsymbol{\delta}) \\ bC_r(\mathbf{x},\boldsymbol{\delta})\end{bmatrix}\end{aligned} \tag{3.23}$$

where

$$\begin{aligned}C_p(\mathbf{x},\boldsymbol{\delta}) &= \Gamma_3 C_l(\mathbf{x},\boldsymbol{\delta}) + \Gamma_4 C_n(\mathbf{x},\boldsymbol{\delta}) \\ C_r(\mathbf{x},\boldsymbol{\delta}) &= \Gamma_4 C_l(\mathbf{x},\boldsymbol{\delta}) + \Gamma_8 C_n(\mathbf{x},\boldsymbol{\delta})\end{aligned} \tag{3.24}$$

3.3 Coordinated Turn

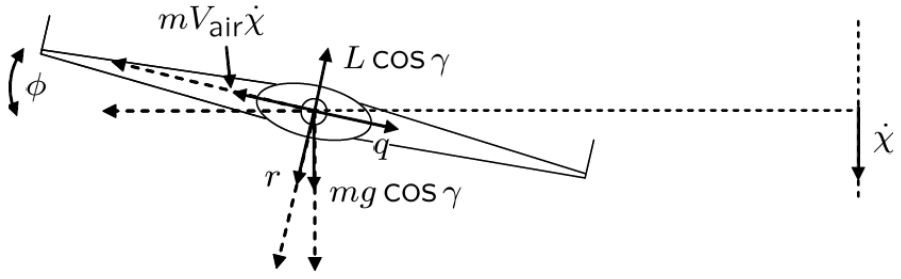


Figure 5: Coordinated Turn Dynamics

A coordinated turn in aviation terminology is a circular turning motion in which the acceleration along the x^b axis is zero. In other words, in a level turn and absence of sidewind, in order for the airframe to perform a turn with radius R, the following must be true:

$$\begin{aligned} L\cos\phi &= mg \\ mV_a\dot{\psi} &= L\sin\phi \end{aligned} \tag{3.25}$$

Combining and solving for $\dot{\psi}$ we get

$$\dot{\psi} = \frac{g}{V_a}\tan\phi \tag{3.26}$$

Thus, we get a relationship between roll angle and desired heading, in order to perform a coordinated turn.

3.4 Extraction of transfer functions

In the following section, an effort is made to extract linear transfer functions between the aircraft inputs and outputs. This is important, because in order to exert PID control, the system under control must be linear.

Realistically, the fixed wing airframe isn't a linear system at all, but under the constraint that it will deviate very little from level flight, valid approximations can be made, and the residual error can be considered as a disturbance that the PID controller can reduce.

Roll

We start by rearranging (3.2) into

$$\begin{aligned} \dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ &= p + d_{\phi 1} \end{aligned} \tag{3.27}$$

$d_{\phi 1}$ is treated as a residual term.

Differentiating and substituting (3.23) onto (3.27) we get

$$
\begin{aligned}
\ddot{\phi} &= \dot{p} + \dot{d}_{\phi 1} \\
&= \Gamma_1 pq - \Gamma_2 qr + \bar{q}Sb\left(C_{p_0} + C_{p_\beta}\beta + C_{p_p}\frac{bp}{2V_a} + C_{p_r}\frac{br}{2V_a} + C_{p_{\delta_a}}\delta_a + C_{p_{\delta_r}}\delta_r \right) + \dot{d}_{\phi 1} \\
&= \Gamma_1 pq - \Gamma_2 qr \\
&\quad + \bar{q}Sb\left(C_{p_0} + C_{p_\beta}\beta + C_{p_p}\frac{b}{2V_a}(\dot{\phi} - d_{\phi 1}) + C_{p_r}\frac{br}{2V_a} + C_{p_{\delta_a}}\delta_a + C_{p_{\delta_r}}\delta_r \right) + \dot{d}_{\phi 1} \\
&= \left(\bar{q}SbC_{p_p}\frac{b}{2V_a} \right)\dot{\phi} + \left(\bar{q}SbC_{p_{\delta_a}} \right)\delta_a \\
&\quad + \left(\Gamma_1 qp - \Gamma_2 qr + \bar{q}Sb\left(C_{p_0} + C_{p_\beta}\beta - C_{p_p}\frac{b}{2V_a}d_{\phi 1} + C_{p_r}\frac{br}{2V_a} + C_{p_{\delta_r}}\delta_r \right) + \dot{d}_{\phi 1} \right) \\
&= \alpha_{\phi 1}\dot{\phi} + \alpha_{\phi 2}\delta_\alpha + d_{\phi 2}
\end{aligned}
\tag{3.28}
$$

Thus, we have expressed the derivative of the roll angle as a linear system with d_a as the input and $d_{\phi 2}$ as a disturbance term that gets larger as the airframe deviates from level flight.

For shallow bank and pitch angles $d_{\phi 1}$ is very close to 0. We can view $d_{\phi 2}$ as residual aerodynamic and inertial effects. Indeed, it contains, among others, the off-diagonal inertial term J_{xy} and the aerodynamic coefficients C_p, other than C_{p_α}.

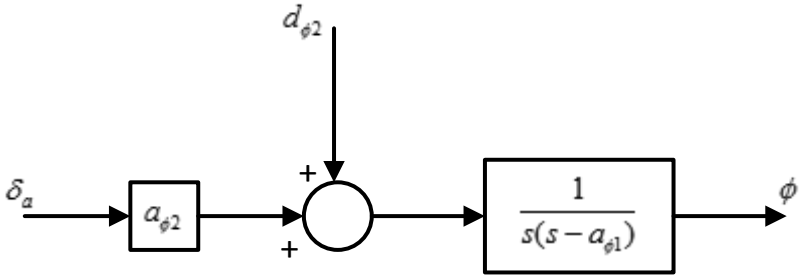


Figure 6: Aileron-to-Roll System

Yaw

Starting from (3.26), we rearrange:

$$
\begin{aligned}
\dot{\psi} &= \frac{g}{V_a}\phi + \frac{g}{V_a}(\tan\phi - \phi) \\
&= \frac{g}{V_a}\phi + \frac{g}{V_a}d_\psi
\end{aligned}
\tag{3.29}
$$

Thus, we have expressed the derivative of the yaw angle as a linear system in terms of the roll angle ϕ and a geometrical residual d_ψ. d_ψ represents the roll approximation error and for shallow bank angles is close to 0.

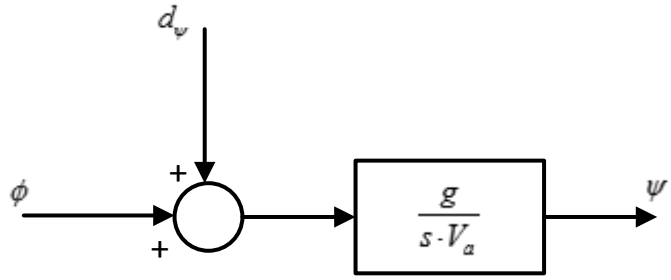


Figure 7: Roll-to-Yaw System

Pitch

From (3.2) we rearrange:

$$\begin{aligned}\dot{\theta} &= q\cos\phi - r\sin\phi \\ &= q + q(\cos\phi - 1) - r\sin\phi \\ &= q + d_{\theta 1}\end{aligned} \tag{3.30}$$

Differentiating and substituting with (3.23) we get

$$\begin{aligned}\ddot{\theta} &= \dot{q} + \dot{d}_{\theta 1} \\ &= \Gamma_6(r^2 - p^2) + \Gamma_5 pr + \frac{\overline{qc}S}{J_y}\left(C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{\overline{c}}{2V_a}q + C_{m_{\delta_e}}\delta_e\right) + \dot{d}_{\theta 1} \\ &= \Gamma_6(r^2 - p^2) + \Gamma_5 pr \\ &\quad + \frac{\overline{qc}S}{J_y}\left(C_{m_0} + C_{m_\alpha}(\theta - \gamma) + C_{m_q}\frac{\overline{c}}{2V_a}(\dot{\theta} - d_{\theta 1}) + C_{m_{\delta_e}}\delta_e\right) + \dot{d}_{\theta 1} \\ &= \left(\frac{\overline{qc}S\overline{c}C_{m_q}}{J_y V_a}\right)\dot{\theta} + \left(\frac{\overline{qc}SC_{m_\alpha}}{J_y}\right)\theta + \left(\frac{\overline{qc}SC_{m_{\delta_e}}}{J_y}\right)\delta_e \\ &\quad + \left(\Gamma_6(r^2 - p^2) + \Gamma_5 pr + \frac{\overline{qc}S}{J_y}\left(C_{m_0} - C_{m_\alpha}\gamma - C_{m_q}\frac{\overline{c}}{V_a}d_{\theta 1}\right) + \dot{d}_{\theta 1}\right) \\ &= a_{\theta 1}\dot{\theta} + a_{\theta 2}\theta + a_{\theta 3}\delta_e + d_{\theta 2}\end{aligned} \tag{3.31}$$

Once again, a quasi-linear system emerges, describing the pitch angle as the output of a system with input δ_e and disturbance $d_{\theta 2}$.

$d_{\theta 1}$ represents the correction of θ regarding the roll angle ϕ. It is close to 0 for shallow bank angles.

$d_{\theta 2}$ term represents the inertial coupling of the body axes as well as the aerodynamic effects on the airframe caused by off-axis quantities.

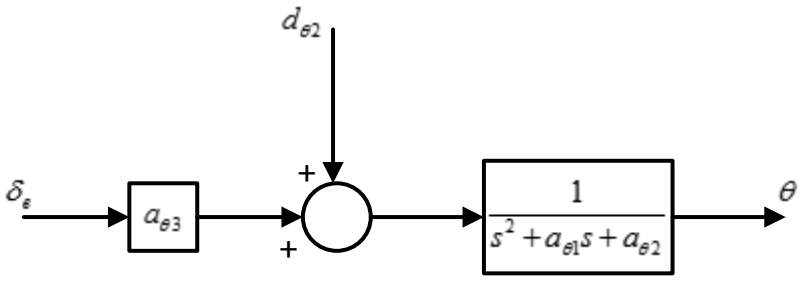


Figure 8: Elevator-to-Pitch System

Altitude

Here we try to derive a description of the altitude in regard to the pitch angle and the airspeed.

From (3.3) we rearrange:

$$\begin{aligned} \dot{h} &= u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta \\ &= V_a\theta + (u\sin\theta - V_a\theta) - v\sin\phi\cos\theta - \cos\phi\cos\theta \\ &= V_a\theta + d_h \end{aligned} \tag{3.32}$$

Depending on the control variable of choice, we can maintain either V_a or θ constant and perform control with the other variable.

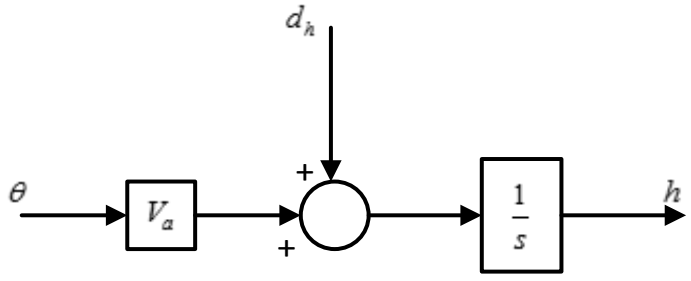


Figure 9: Pitch-to-Altitude System

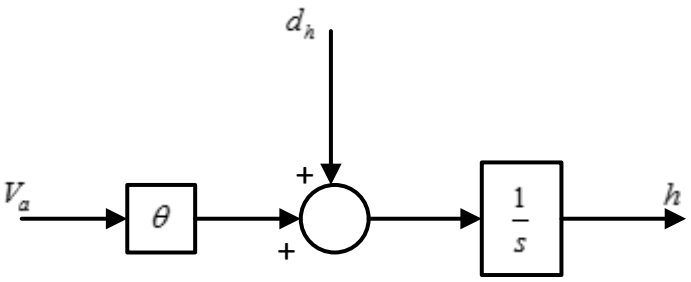


Figure 10: Airspeed-to-Altitude System

Airspeed

Finally, we extract a transfer function from throttle and pitch angle to the resulting airspeed.

Assuming that the wind speed is zero, then the airspeed magnitude is

$$V_a = \sqrt{(u^2 + v^2 + w^2)} \tag{3.33}$$

Differentiating, we get:

$$\dot{V_a} = \frac{u(\dot{u}) + v(\dot{v}) + w(\dot{w})}{V_a} \tag{3.34}$$

Taking into account that the airspeed components in the body frame are expressed as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = V_a \begin{bmatrix} \cos\alpha\cos\beta \\ \sin\beta \\ \sin\alpha\cos\beta \end{bmatrix} \tag{3.35}$$

Combining (3.34) and (3.35) we get

$$\begin{aligned} \dot{V_a} &= \dot{u}\cos\alpha\cos\beta + \dot{v}\sin\beta + \dot{w}\sin\alpha\cos\beta \\ &= \dot{u}\cos\alpha + \dot{w}\sin\alpha + d_{V_1} \\ d_{V_1} &= -\dot{u}(1-\cos\beta)\cos\alpha - \dot{w}(1-\cos\beta)\sin\alpha + \dot{v}\sin\beta \end{aligned} \tag{3.36}$$

Notice that for small sideslip angles d_{V1} goes to 0.

Also, using (3.4) and (3.22) we expand into

$$\begin{aligned} \dot{V_a} = \cos\alpha & \left(\begin{array}{l} rv - qw - g\sin\theta + \frac{\bar{q}S}{m}\left(C_{X_0} + C_{X_\alpha}\alpha + C_{X_q}\frac{\bar{c}}{2V_a}q + C_{X_{\delta_e}}\delta_e \right) \\ + \frac{1}{2m}\rho S_{prop}\left((k\delta_t)^2 - V_a^2 \right) \end{array} \right) \\ + \sin\alpha & \left(\begin{array}{l} qu - pv + g\cos\theta\cos\phi \\ + \frac{\bar{q}S}{m}\left(C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}\frac{b}{2V_a}p + C_{Y_r}\frac{b}{2V_a}r + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r \right) \end{array} \right) + d_{V1} \end{aligned} \tag{3.37}$$

Then simplifying with (3.17)

$$
\begin{aligned}
\dot{V_a} &= \left(rV_a\cos\alpha - pV_a\sin\alpha\right)\sin\beta - g\sin(\theta-\alpha) - g\sin\alpha\cos\theta(1-cos\phi) \\
&+ \frac{\bar{q}S}{m}\left(\left(C_{D_0} + C_{D_\alpha}\alpha + C_{D_q}\frac{\bar{c}}{2V_a}q + C_{D_{\delta_e}}\delta_e\right)\right) \\
&+ \frac{1}{2m}\rho S_{prop}\left((k\delta_t)^2 - V_a^2\right)\cos\alpha + d_{V1} \\
&= -g\sin\gamma + \frac{\bar{q}S}{m}\left(\left(C_{D_0} + C_{D_\alpha}\alpha + C_{D_q}\frac{\bar{c}}{2V_a}q + C_{D_{\delta_e}}\delta_e\right)\right) \\
&+ \frac{1}{2m}\rho S_{prop}\left((k\delta_t)^2 - V_a^2\right) + d_{V2}
\end{aligned}
\tag{3.38}
$$

$$
\begin{aligned}
d_{V2} &= \left(rV_a\cos\alpha - pV_a\sin\alpha\right)\sin\beta - g\sin\alpha\cos\theta(1-\cos\phi) \\
&+ \frac{1}{2m}\rho S_{prop}\left((k\delta_t)^2 - V_a^2\right)(\cos\alpha - 1) + d_{V1}
\end{aligned}
$$

Once again, when in level flight, d_{V_2}=0.

Since (3.38) is not linear, in regards to γ or δ_t, we need to perform a linearization around the trim point $(\delta_t^*, \theta^*, V_a^*)$ and consider the deviation from that point $(\overline{\delta_t}, \overline{\theta}, \overline{V_a})$. The result, after the linearization is

$$
\begin{aligned}
\dot{\overline{V_a}} &= -g\cos(\theta^* - \alpha^*)\overline{\theta} + \left(\frac{\rho V_a^* S}{m}\left(-C_{D_0} - C_{D_\alpha}\alpha^* - C_{D_{\delta_e}}\delta_e^*\right) - \frac{\rho S_{prop}}{m}V_a^*\right) \\
&+ \left(\frac{\rho S_{prop}}{m}k^2\delta_t^*\right)\overline{\delta_t} + d_V \\
&= a_{V_1}\overline{V_a} + a_{V_2}\overline{\delta_t} + a_{V_3}\overline{\theta} + d_V
\end{aligned}
\tag{3.39}
$$

Thus, we now have a linear formula for the change of the airspeed around trim, with both throttle and pitch angle as input. Depending on which one we want to use as regulating input, we can keep the other constant and perform control over one input variable.

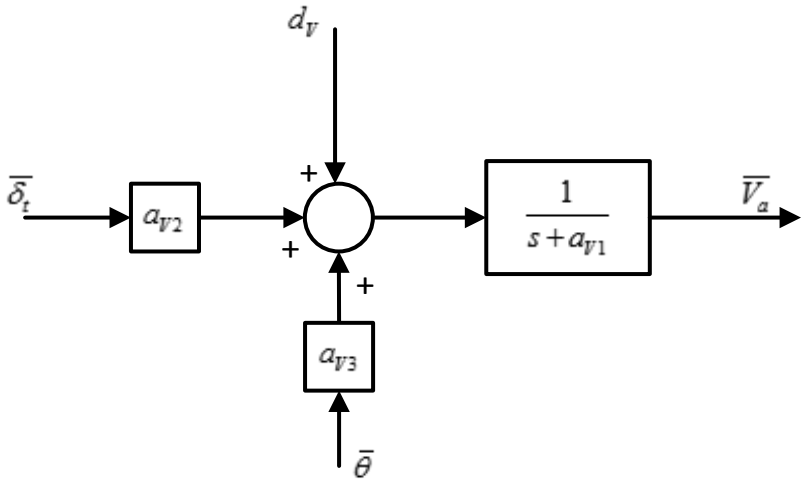


Figure 11: Airspeed System

4 Solution Process

4.1 Navigation Controller

As in most autonomous control systems, the approach that will be used is top-down, while keeping emerging requirements in check, to ensure realistic solutions. In that approach, we will first specify the high-level needs, the features of the Unmanned Aerial System (UAS), and let them dictate the implementation details.

These features mirror the general capabilities of a UAS. Modern UASs can execute a multitude of tasks related to their mission, including but not limited to:

1. Take off
2. Perform as a fly-by-wire system for a human operator
    a. Stabilize the airframe
    b. Ensure collision avoidance
    c. Set up a virtual geo-fence around an available flight area
    d. Limit the lower and higher operational altitude.
    e. Return the aircraft to the launch site on failsafe situations
3. Carry out a flight mission plan
    a. Visit waypoints
    b. Calculate trajectories that join these waypoints, based on various criteria
    c. Loiter around waypoints
    d. Engage tasks based on events
    e. Manipulate their payload
    f. Provide a real-time telemetry downlink to a base station
    g. Maintain a real time two-way communication link with a base station
4. Perform a landing in the launch location or elsewhere.

Since there where constraints both in time and budget, the tasks that were chosen for implementation were:

1. Take off
2. Incorporate fly-by-wire functions
    a. Stabilize the airframe
    b. Return to launch
3. Carry out a flight mission plan
    a. Visit waypoints
    b. Plan a straight-line trajectory between the current aircraft point and the next waypoint
    c. Loiter around waypoints

The abstract control layer responsible for these tasks is called Navigation Controller. This layer is not responsible for commandeering the aircraft. Its purpose is to issue navigation commands in order to complete the flight mission. These commands will be received afterwards by lower control layers which will ensure their execution. The Navigation Controller will consider that these commands will actually be carried out and will not wait for confirmation, reducing interface with lower layers to a minimum. As a result, the Navigation Controller can operate completely independently from lower layers, via an abstract communication channel.

Going into further detail to describe the operation of the Navigation Controller, each aforementioned task is described below, accompanied by a flow chart.

Take off

In this task, the achievement of critical airspeed as fast as possible is the primary objective, whilst preserving a constant heading, in order to align with the airstrip, and level angle of the wings, in order to avoid collision with the ground.

The navigation controller issues commands for maximum thrust, zero roll angle, and constant heading. Heading is controlled with the rudder signal, since it is coupled with the steerable front wheel. In full-scale airplanes, the steerable wheel is actually locked in a straight position, but in the case of our model there is no such option; the coupling in constant. Once take-off speed is achieved, constant climb airspeed is commanded. This task is completed once a preset altitude is achieved.

Summing up, there are two distinct phases in the take-off procedure:

1.Set throttle to full, maintain heading using rudder, maintain level wings

2. Set throttle to full, maintain heading using ailerons, maintain climb airspeed.

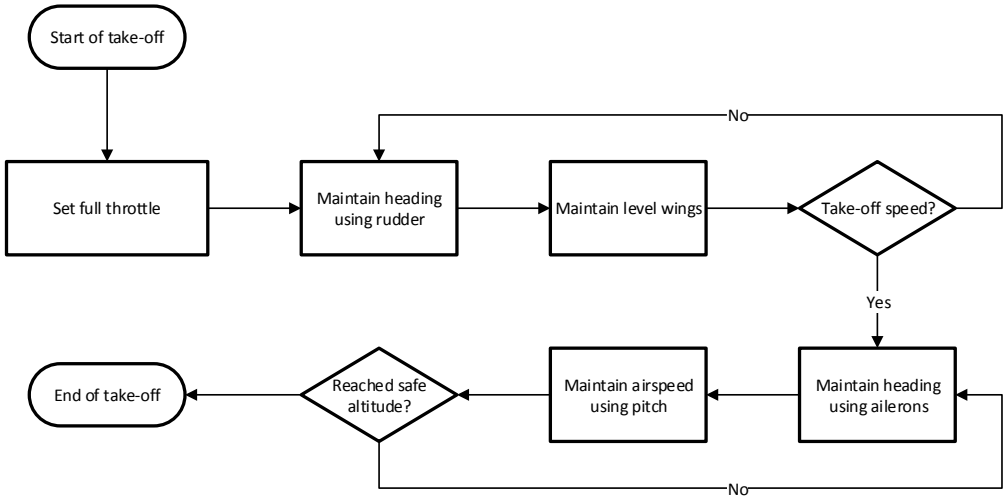


Figure 12: Take-off Procedure

Point to point navigation.

For simplicity and ease of implementation, the Navigation Controller simply finds the heading leading from the current aircraft position to the target waypoint and issues a heading command. The heading is calculated using the great circle earth model (Weisstein, n.d.; Movable Type Ltd, n.d.). This value is recalculated in each software loop.

If a current longitude and latitude of the aircraft is λ_1 and ϕ_1 and the longitude and latitude of the target waypoint are λ_2 and ϕ_2, then the desired heading that leads from point 1 to point 2, at the location of point 1, is

$$\theta = \operatorname{atan2}\left(\sin\left(\frac{\lambda_2 - \lambda_1}{2} \right) \cos\phi_2, \cos\phi_1 \cos\phi_2 - \sin\phi_1 \cos\phi_2 \cos\left(\frac{\lambda_2 - \lambda_1}{2} \right) \right)$$

For small operational areas, such complicated computations may be redundant; instead, treating the map coordinates as points in a Cartesian plane would work. However, in case of a system extension, which would allow it to cover several kilometers in a single flight, then accuracy becomes important to avoid longer-than-needed flight paths, especially in locations with large latitude.

If no other restrictions apply, the Navigation Controller also commands the desired cruising airspeed.

There are several ways to test if the aircraft has reached the waypoint. The easiest, but not the most robust one, is to examine the distance between the aircraft and the waypoint. If this is lower than a threshold (usually a few meters), the waypoint can be considered visited.

The difference in altitude between the two points must be taken in to account while planning a mission; the aircraft cannot climb or descend at an arbitrary angle. Instead, these actions must be confined between two extreme values. The climb angle limit depicts the inability of the aircraft to climb vertically, since it will bleed of airspeed and stall. The descent angle limit ensures that the aircraft will not reach overspeed. That would alter the control surface aerodynamics significantly and controlling the vehicle in that state would be difficult but also dangerous for the airframe integrity.

These limits are not constant. As explained above, the constraining factor is airspeed and thus this should be the defining quantity while setting maximum climb/descend angles. However, regulating the airspeed to avoid stall is not a task of the Navigation Controller.

For the above reasons, in case the climb/descend angle limits inhibit the vehicle from achieving the desired altitude upon arrival at the waypoint, the navigation controller will issue a loitering command (explained below) to reach the target altitude immediately afterwards.

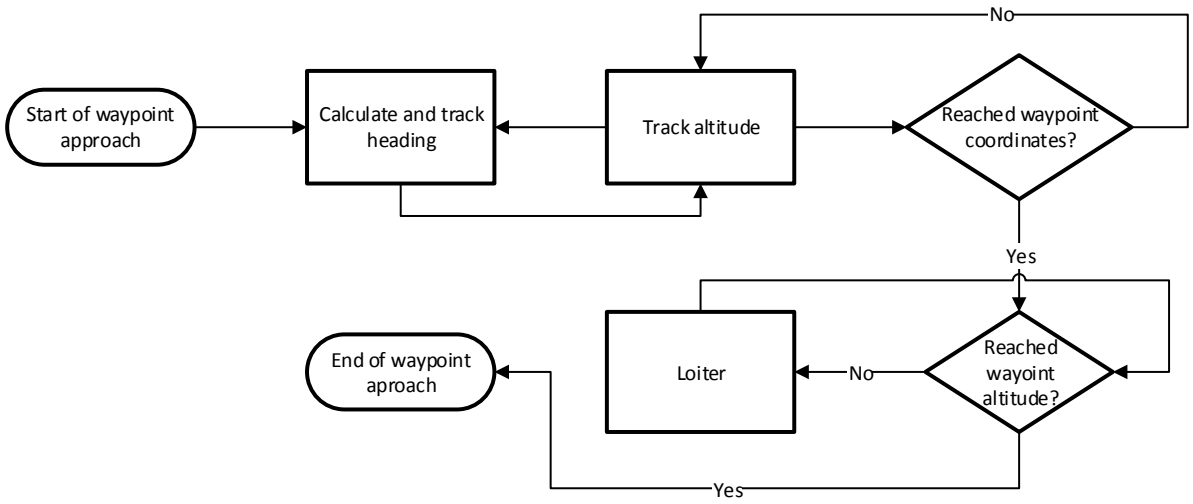


Figure 13: Waypoint Visiting Procedure

Loitering

In autopilot terms, loitering describes a mode where the aircraft is moving in circles above a set point on the ground. The altitude can vary during this mode, allowing for the aircraft to gain or lose height while staying inside a confined column of airspace.

The challenge in this mode is to maintain a steady turning motion and compensate for disturbances that make the vehicle drift away from the circle and draw a spiral moving away from the target. In order to achieve this, a vector field method was used. This method calculates the desired heading angle as a function of the distance (radius)

of the vehicle from the target. This method also provides a smooth entry in the loitering pattern.

The formula for the desired heading at all times is:

$$yaw_d = \gamma - \frac{\pi}{2} - \tan^{-1}\left(k\left(\frac{d}{r} - 1\right)\right)$$

Where: γ is the angle of the direction vector pointing from the loiter center to the aircraft

d is the distance of the aircraft from the loiter center

r is the desired loiter radius

k is an positive constant that adjusts the transition into the circle

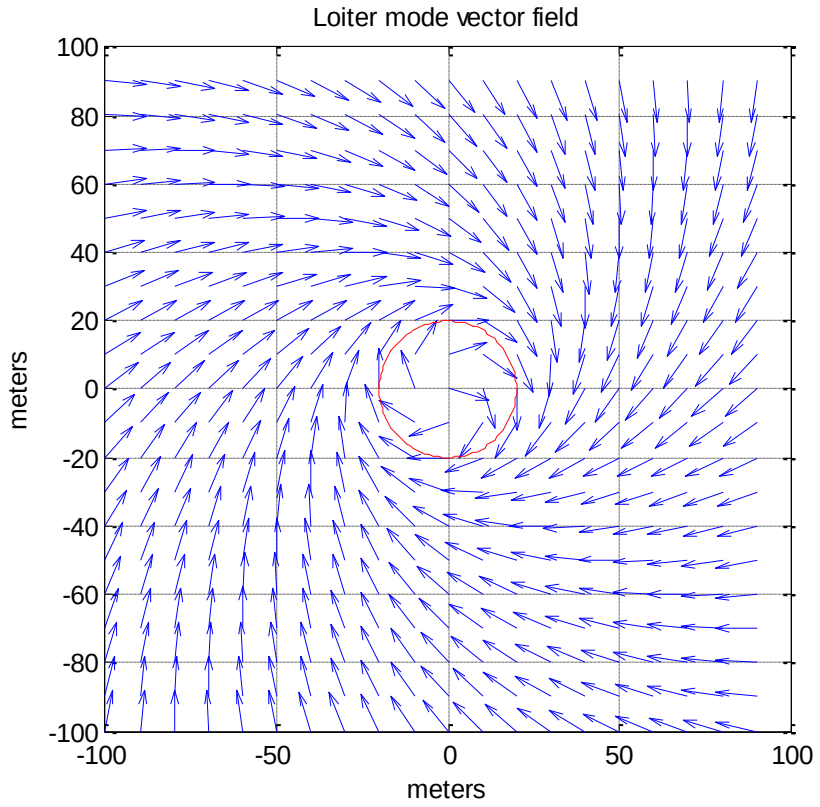


Figure 14: Loiter Vector Field

Figure: Vector field for loiter pattern around (0,0) and radius of 20 meters

The distance between the aircraft and the loitering center is once again by the Greater Circle Earth model:

$$d = R_{Earth} \cdot c$$

Where

$$c = 2\text{atan2}(\sqrt{a}, \sqrt{1-a})$$
$$a = \sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos\phi_1 \cos\phi_2 \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)$$

Layer Inputs/Outputs

Based on the above, it becomes evident that the Navigation Controller must be able to have knowledge of the following aircraft quantities:

1. Map coordinates
2. Altitude

And be able to command the following aircraft quantities:

1. Heading
2. Altitude
3. Airspeed

4.2 Motion Controller

Continuing the top-down approach, we will now formulate the next control layer, the Motion Controller. This layer is responsible for executing the navigation commands issued from the Navigation Controller. In other words, the Motion Controller must be able to regulate the aircraft heading, altitude and airspeed. However, it is independent of the Navigation Controller implementation and algorithms.

At this point we must decide the type of the controllers we will use. Since the objective of the thesis was the complete integration of the system, the Motion Controller was selected to be simple and proven. Thus, PID controllers were chosen.

The next decision was to decide which inputs would be used to control which quantities. In this approach, lateral variables are used to control lateral quantities and longitudinal variables for longitudinal quantities. Essentially, in accordance with the previous chapter, we will use the lateral/longitudinal magnitudes decoupling in our favour, in order to design simpler controllers.

Aircraft Dynamics

As was demonstrated in the previous chapter, it is clearly visible that unless we have all of the aerodynamic coefficients and the inertial matrix, an extraction of an analytical dynamic model is impossible. This is why an effort was made to come up with that model experimentally.

In the previous chapter, it was observed that the system could be approximated by transfer functions, each spanning on a single axis on the body frame. As a result, these transfer functions are not supposed to be coupled with each other. Numerical models were estimated via input/output measurement pairs for the roll, pitch and yaw axis as well as the thrust model independently. This was done in the Model Estimation Toolbox of the Matlab suite.

In detail, for each control input (aileron, elevator, rudder, throttle), while the aircraft was in level flight (trim condition), the control signal was fluctuated from trim position, to minimum, then to maximum and finally back to trim position. The response of the corresponding axis was recorded, as well as the input control signal.

This procedure was repeated three times for each axis. The input-output pairs where introduced into the Model Estimation Toolbox environment. The first two where combined into an experiment to train the model estimator. The last one was used to evaluate the extracted system.

Finally, the toolbox was allowed to choose between ARMAX models of various orders and the one with the best fit was selected to represent that axis.

In retrospect, it should be noted that the model estimation procedure failed and the controller gains that resulted from that model were extremely weak and unusable. The gain tuning procedure was done manually in the end and is described in a following paragraph.

PID Control Loops

Even if we don't have a numerical model for the system dynamics, their theoretical expression is well described in chapter 4. Based on those models, we continue on designing the PID control loops, even though their gains will be will not be calculated theoretically.

Lateral Control

The control loops that were formed for the lateral motion are:

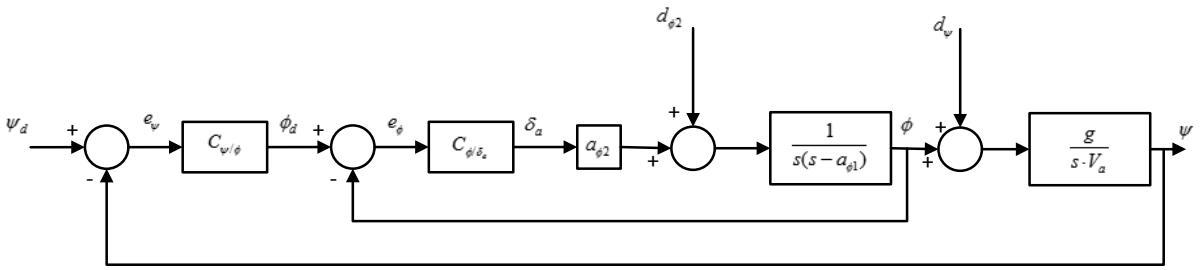


Figure 15: Lateral Autopilot

And incorporate the PID controllers:

- Yaw-to-Roll
- Roll-to-Aileron

Longitudinal Control

The longitudinal controller is tasked with regulating both altitude and airspeed. As discussed previously, those two quantities are interdependent dynamically, so the following step-wise controller regime is used

The following graphic showcases that control regime.

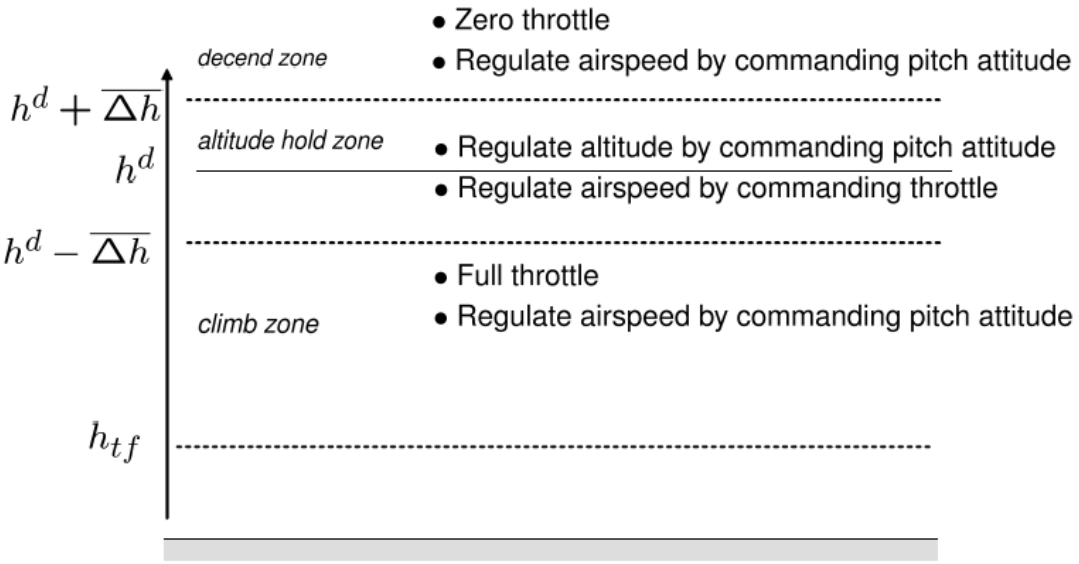


Figure 16: Altitude Control Scheme

The primary objective is the regulation of altitude. The possible altitude range is split into four sectors.

For any altitude below take-off altitude h_{tf}, the take-off controller regulated throttle and airspeed and is discussed previously.

Around the desired altitude h_d, a zone is defined, in which we are content with the aircraft flying, while having a maximum altitude error Δh. In this zone, we need to establish a cruising speed and this is achieved with the use of throttle. Also, in order to stay within this zone, we make use of the elevator to control the altitude.

If the aircraft is lower than Δh from the target altitude, it needs to climb in order to reach the altitude hold zone. To do that, the throttle is set to full and the pitch angle regulates the airspeed, in order to avoid stalling.

If the aircraft is higher than Δh from the target altitude, it needs to descend in order to reach the altitude hold zone. To do that, the throttle is set to zero and the pitch angle regulates the airspeed, in order to avoid overspeed.

In short, when the aircraft is much lower than its target altitude, it climbs at a constant safe minimum airspeed. When it is much higher than its target altitude, it glides at a safe maximum airspeed. When it is close to the target altitude, in regulates the altitude with the pitch angle while maintaining cruising speed.

To perform the above controls we need the following controllers:

- A step-wise controller that will switch between the three control modes.
- A controller able to regulate the pitch angle via the elevator

- A controller able to regulate airspeed via pitch angle
- A controller able to regulate airspeed via throttle

Consequently, we form the airspeed and altitude control loops:

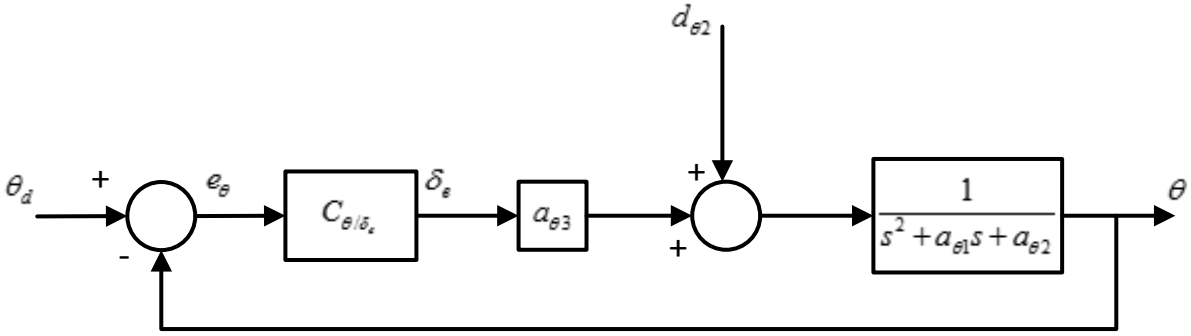


Figure 17: Pitch Hold Controller

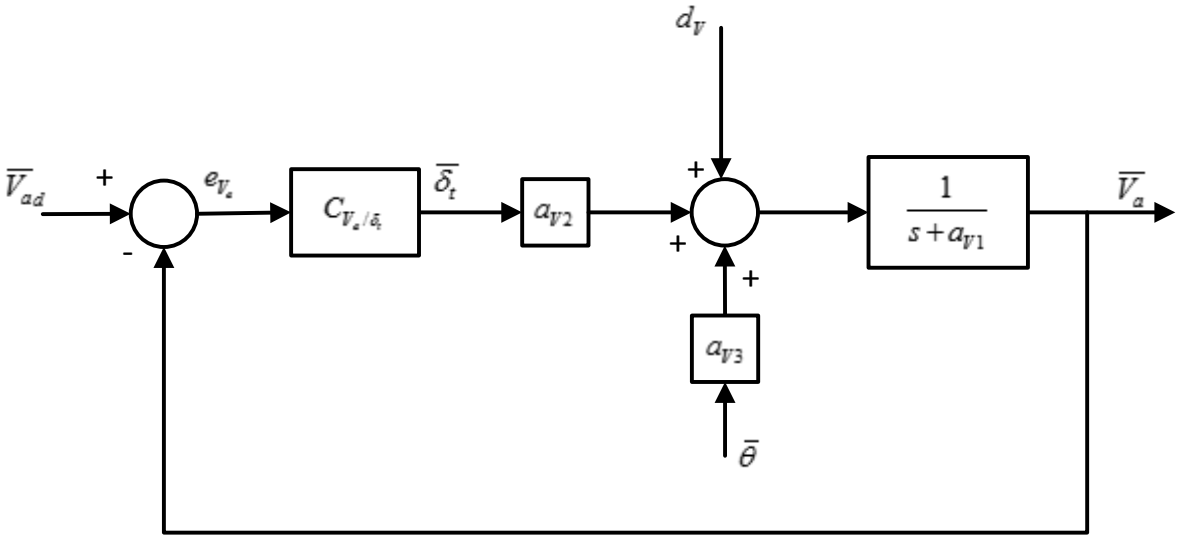


Figure 18: Throttle-to-Airspeed Controller

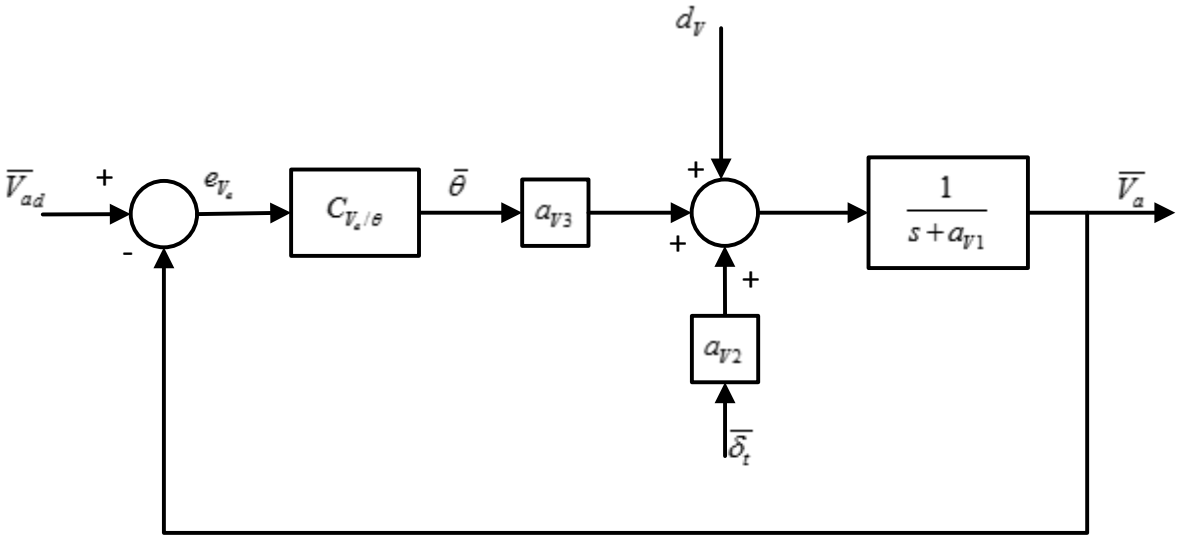


Figure 19: Pitch-to-Airspeed Controller

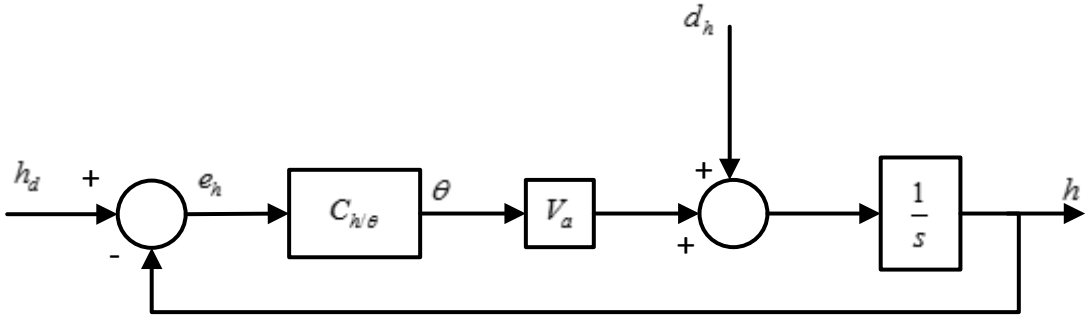


Figure 20: Pitch-to-Altitude Controller

Notice that the rudder isn't being used during flight. Generally in aviation, rudder is used for coordinated turns and for trimming sideslip caused by lateral winds. For small model aircraft, while this type of control is applicable, isn't of great importance.

Take Off Control

All the Motion Controller components needed to perform take off are shared with the previous functions, as described in the corresponding Navigation Controller section, except for the heading control while the aircraft is on the ground.

A controller for the rudder and front wheel is needed, in order to ensure that the aircraft is tracking straight during take off.

The corresponding PID-type controller should include an I term, since very often the driving wheel of the aircraft is offset (due to manufacturing errors) in relation to the

rudder center position. Not including the I term would render the controller unable to reject that step-type disturbance.

The controller should be extremely smooth while driving the aircraft on the ground; any sudden movements, especially while the aircraft has built up enough ground speed, can topple it. Usually the 3-wheeled undercarriage doesn't provide a wide support area.

PID Controller tuning

In order to come up with the controller gains, initially, Matlab was once again used: the controllers were inserted in their respective control loops and the gains were automatically calculated based on the numerical models estimated previously and the desired controller response.

The function of the digital PID controller that Matlab implements is:

$$C(z) = P + I \cdot T_s \frac{1}{z-1} + \mathrm{D} \frac{N}{1 + N \cdot T_s \frac{1}{z-1}} \tag{3.40}$$

Where: P is the proportional gain

I is the integral gain

D is the differential gain

N is the low-pass filter coefficient

Ts is the sampling interval

Unfortunately, in practice, those controllers were proven to be too weak, especially for the Yaw-to-Roll and Pitch-to-Elevator, probably due to the simplistic approach in the modeling process. Steady state error or even divergence was observed.

In the end, as a last resort, manual tuning through live observation was used. The onboard computer was programed with a routine that allowed gain tuning in-flight. The gains were incrementally adjusted and set as to produce a relatively smooth and visually pleasing result.

It should be noted, however, that in order to preserve a relatively fast system response, some oscillations were preserved. In the end this tuning procedure was deemed as very crude and didn't produce satisfactory results. However, it was the only one that provided operational gains, at least up to a degree.

Layer Inputs/Outputs

Closing, as before, we will sum up the necessary input quantities for the Motion Controller:

1. Orientation
2. Airspeed
3. Altitude

It also must be able to communicate the following quantities to the hardware layer:

1. PWM signals of the aileron, elevator, rudder servos and the motor ESC.

4.3 Hardware

As in any robotic system, sensors are needed to provide the necessary feedback and give the robot spatial cognition. Fortunately, advances in Micro Electro Mechanical Sensors (MEMS) have produced very light, small and cheap hardware. Of course, there still is a very wide price range, reflecting the differences in accuracy and quality among products. The trade-offs of low cost MEMS sensors are moderate precision and most importantly lack of factory error compensation. This translates to significant offset and scaling errors that should be compensated for before the integration of the sensors in the module.

Major factors in the hardware configuration are the aerial platform where it will be installed and the available budget. While the aircraft to be used isn't presented yet, a reasonable guideline is to keep the total autopilot weight under 500gr. This keeps the airframe wingspan to under 1.5m typically. The lighter the autopilot, the smaller the aircraft it can be installed on. Cost should be kept to a few hundred euros, to reflect the amateur market the product is intended for.

While setting up the hardware, we will start by making sure we have an array of sensors that are able to provide the system with the required measured quantities. Below are listed the aforementioned quantities and the sensors that provide them. They all are cheap off-the-shelf products. This displays the robustness of the solution and its low requirements in terms of hardware quality.

Global Position

As expected, a simple GPS receiver unit can provide knowledge of the position of the aircraft in longitude and latitude. Additionally, GPS receivers can provide a wide supplementary set of information, such as (Fastrax Ltd, 2010):

1. Altitude

2. Ground speed
3. Heading
4. Number of satellites tracked
5. Date and time

In basic GPS systems, where the reference signal in solely provided by the constellation of satellites visible by the receiver, there is a typical error in the position reading of up to 5 meters. While this sounds large, if we consider that the scale of a flight mission goes up to the kilometer range, its importance diminishes.

Besides, most modern receivers can incorporate DGPS correction signals from the SBAS system. In Europe, these are part of the EGNOS system and are provided by dedicated satellites. With this simple software option, horizontal accuracy can be improved to the 2.5 meter range. Unfortunately, no further improvement is possible without additional hardware (European Space Agency, 2013).

The vertical accuracy of the position reading, in other words altitude, lies in the range of 10 meters, due to the geometry of the satellite constellation.

Typical fix frequency is 1-20Hz, which doesn't allow control of the aircraft solely based on the GPS readings. Additionally, we should keep in mind that reading error isn't constant within a short time frame; the returned position can lie anywhere within the error sphere around the actual receiver position.

This has consequences on the Ground Speed and Heading readings. Since these are constructed by differentiating recent position fixes, they can be very erratic and generally unreliable and only indicative.

Based on the above information, we come to the conclusion that the horizontal accuracy is of adequate resolution in space, altitude can't be used for small elevation corrections and Ground Speed and Heading readings are unreliable.

The sensor of choice was the UP501 by Fastrax. It operates on 3.3V and can provide an SBAS DGPS reading at a frequency of 1Hz. It communicates with a Serial interface with variable baud rate. It is a low price choice, common in amateur robotic systems, and boasts small size and weight. It has a ceramic patch antenna which should be pointed towards the sky at all times.

Figure 21: The GPS Module

Altitude

While the GPS receiver provides information for altitude, as we saw before, this measurement is unreliable and not available on a timely manner. For those reasons, we use a secondary device for altitude measurement: a barometer.

Barometric pressure varies with altitude in a predictable manner, as can be seen in the following formula, which holds for low altitudes below 11km (Wikipedia contributors, 2013):

$$P = P_0 \left(\frac{T_0}{T_0 + L_0 h} \right)^{\frac{g_0 M}{R^* L_0}}$$

Where P_0 is the atmospheric pressure at sea level

T_0 is the temperature at sea level

L_0 is the temperature lapse rate below 11km

g_0 is the gravity acceleration

R^* is the universal gas constant

M is the molar mass of the air

However, as the reader can see, temperature must also be taken into account. This is why most barometers are usually operated along with a thermometer. Thus, temperature reading is incorporated in the altitude calculation and temperature compensation is executed.

In practice, the MEMS sensor will also have a temperature drift error, which is another reason to incorporate a thermometer in the system. Also, usually the barometer sensor

manufacturer will provide algorithms and approximations to calculate both true barometric pressure and barometric altitude. Thus, in practice, theoretical formulas aren't used as-is.

A typical barometer accuracy in altitude is in the range of 0.5m. Better performance can be achieved by averaging measurements over time, but this sacrifices time for performance.

If one wants to measure absolute altitude (in regard to sea level), he must know the barometric pressure at sea level in the approximate area at that time. While this information is usually available on forecast internet sites, it may not be accessible on the field. Instead, a useful approach is to operate based on Absolute Ground Altitude (AGL). The 0 reference altitude is sampled when the system is initialized and it is deducted from all following measurements, providing an accurate AGL reading in every mission.

The device of choice was the BMP085 sensor by Bosch, which combines a barometer and a thermometer in the same die. It is located on a Break-Out-Board by 3DRobotics. The supply voltage was selectable with a solder jumper between 5V and 3.3V. The device communicates with I2C protocol.

Figure 22: The Barometer Module

Airspeed

In every aviation application, the measuring of airspeed is executed by differential pressure sensors, attached on Pitot tubes. A Pitot tube is pair of coaxial straight length of metal tubing with a pinhole in the front end and a few similar pinholes on the side of the tubing, a short distance away from the front. These pinholes allow air to enter the two separate tubes. The Pitot tube is aligned parallel to the aircraft longitudinal axis during operation.

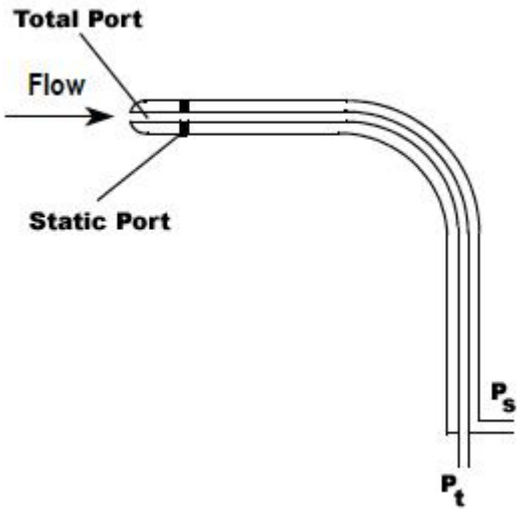


Figure 23: Pitot Tube Representation

Air that enters through the front pinhole corresponds to the fast travelling air (stagnation/total pressure), relative to the aircraft. The side pinholes allow ambient air to fill the outer tube (static pressure). The two tubes lead to the back side of the Pitot tube to two pressure outlets. A differential pressure measurement related to the airspeed can be sampled from those outlets. Silicon tubing is attached on those outlets and on the other side to the sampling port of a differential pressure sensor. That sensor translates the mechanical force of pressure to electrical output. However, the differential pressure measurement must still be converted to airspeed in the microprocessor. Bernoulli's equation is used for this (Houghton & Carpenter, 2003):

$$p_t = p_s + \left(\frac{\rho V^2}{2} \right) \Leftrightarrow$$

$$V = \sqrt{\frac{2(p_t - p_s)}{\rho}}$$

Where p_t is the total pressure

p_s is the static pressure

ρ is the air density

V is the air velocity

However, air density is a function of atmospheric pressure and temperature, thus once again the system uses atmospheric pressure measurements which are temperature compensated.

Special care must be applied to the mounting of the Pitot tube. It must be placed carefully parallel to the aircraft longitudinal axis and have access to clean, undisturbed airflow. Thus, it must not be placed in the area behind a propeller nor very close to the

wing surfaces. Popular mounting positions are on the nose, for aircraft without a single tractor motor, or in each wingtip. Each Pitot tube manufacturer posts his own mounting directions. For our specific hardware, the static pressure pinholes should be placed at least 1cm in front of the leading edge of the wing.

The sensor of choice was the MPXV7002DP by Freescale .It has a differential pressure range of -3kPa to +3kPa. During flight, since the aircraft can move only forward, only positive measurements occur. This range roughly translates to airspeeds up to 200km/h, more than enough for every ordinary model airplane. The sensor is powered by 5V and outputs analog voltage readings.

Figure 24: The Airspeed Measurement Kit

Orientation - Euler Angles

This might as well be the most important sensor measurement system in our system. It allows feedback for almost every orientation control, which is the main function of the autopilot. The UAS simply cannot exist without orientation measurement.

Unfortunately, unlike all the previous sensor types, dedicated mechanical inclinometers don't scale down and are not available in small scale aircraft. Another method must be used to obtain the Euler angles. Sensor fusion systems exist that combine other kinematic variables to come up with a solution set for the orientation. Those systems are called Attitude and Heading Reference Systems (AHRS) and come in a few forms.

A popular approach is to use acceleration and compass data. This is mostly used in cell phones for simple orientation problems. However, it only provides correct steady state solutions. Any dynamic motion throws off the accelerometer readings and renders this approach useless in moving platforms.

Another method uses GPS differential data, accelerometer and gyroscope readings, in order to converge to a solution. This provides a valid solution set, but has the drawback of relying on unreliable GPS readings or, even worse, on the existence of GPS signal

in order for the algorithm to not degrade to a dead reckoning one. Another disadvantage is that the aerial platform must be moving to obtain valid GPS differential data.

Finally, one can combine acceleration, gyroscope and magnetometer data to estimate the solution. In this thesis, this option is used and the algorithm is based on the work of Seb Madgwick and the relatively recent paper he published (Madgwick, Harrison, & Vaidyanathan, 2011). In contrast to other methods, it is not affected by linear accelerations produced in the body frame, described by the Corriolis equation, during turning motions nor does it need differential GPS data to find a solution, meaning that it can perform even when the airframe is still. However, it should be noted that this isn't the only working algorithm on autopilots for model aircraft based on this data set.

The AHRS algorithm of S. Madgwick performs a steepest descent minimum solution approach, in combination with dead reckoning. It uses the constant gravity vector measured from the accelerometer and the constant magnetic field of the Earth measured from the magnetometer to converge to the solution with step size dictated by the gyroscope. However, it needs to estimate the solution at a rate faster than the solution is varying in time. Typical required iteration speed can be as slow as the fastest change of orientation, according to the algorithm author. In this thesis, it is performed at about one order of magnitude faster than the change of orientation.

S. Madgwick also advertises the internal elimination of soft iron magnetic distortions of the magnetometer sensor. However, for security, full calibration of the sensor after installation is performed.

Another advantage of this algorithm is that it formulates the solution in quaternions and thus avoids singularities (gimbal locks). For the sake of interfacing, the quaternion solution vector can then be converted into Euler angles.

In any case, the estimated yaw angle is referenced to the magnetic North. To obtain the geographic North, the Declination angle must be taken into account. In the general area of Greece, this is approximately 5 degrees eastward (World Data Center for Geomagnetism, 2010).

Below is described the sensor selection for the sensor fusion system:

Accelerometer/Gyroscope

The MPU6050 module by InvenSense was used, placed on the corresponding Break-Out-Board by Sparkfun. This module incorporates both an accelerometer and gyroscope sharing the same tri-orthogonal measurement reference system. This ameliorates the measurement quality.

The device is powered by 3.3V and communicates via an I2C protocol. A multitude of set-up options are available that allow selection of maximum measurement rates, sampling rate and low-pass filtering.

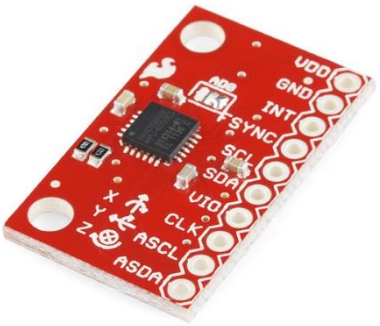


Figure 25: The Inertial Measurement Unit

Calibration and compensation is mandatory when incorporating low-cost MEMS sensors in a system:

The accelerometer was compensated for bias offset errors. Each axis was placed vertically, so that the other two axes recorded zero acceleration. The bias offset against nominal 1G acceleration was recorded and then deducted from all consequent measurements. No scaling error was taken into account.

The gyroscope was also compensated only for bias offset errors. The device was left still for a period of time and measurements in all 3 axes were recorded and averaged. Then, the result was deducted from all consequent measurements. No scaling error was taken into account.

The magnetometer device of choice was the HMC5883L by Honeywell placed on the corresponding Break-Out-Board sold by Sparkfun. It is powered by 3.3.V and communicates via an I2C port. Calibration and compensation of the magnetometer should be much more involved than those of the previous sensors (Freescale, 2012; Freescale, 2013).

Figure 26: The Magnetometer Unit

The locus of the magnetometer sensor, while measuring the magnetic field of the Earth should be a sphere centered on the origin with a radius of 450mT approximately (NOAA, National Geophysical Data Center, 2013). This magnetic intensity varies little over the surface of the Earth and is available from online data.

Magnetometers suffer from two types of errors, on top of bias offset and drift errors, common to all MEMS sensors: Hard magnet and soft magnet errors.

Hard magnet errors refer to bias offset error vectors introduced to the sensor by nearby electrical currents. Sources of such constant magnetic error vectors are power wires in the vicinity of the sensor, including power traces in the autopilot PCB. These errors translate the spherical locus away from the origin by that constant error vector.

Soft magnet errors are more complex. They are the result of distortion in the magnetic field density caused by metallic objects in the vicinity of the sensor. As the sensor is rotated, the magnetic field lines pass through those metallic objects in a different angle, resulting in a change in measured field intensity dependent on the orientation. The measurement locus is reshaped into an ellipsoid that can even have its axes rotated. This is much harder to compensate for.

The calibration and compensation approach used is as follows:

While the sensor was placed in the final hardware configuration, the aircraft was rotated in all directions and 1000 samples of the magnetic vector were taken and imported in the Matlab suite.

The result can be visualized in the following pictures.

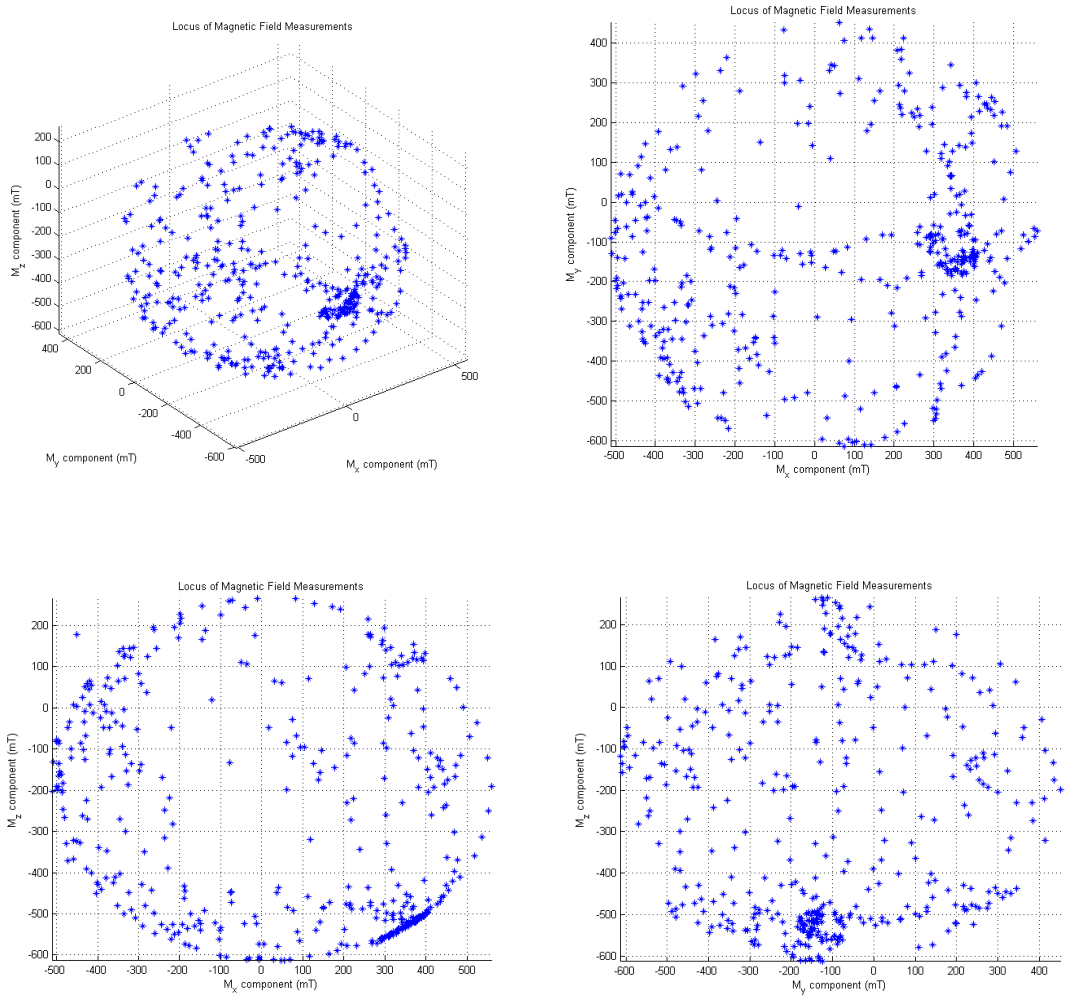


Figure 27: Uncompensated Magnetic Measurements

The constant offset is clearly visible, as well as the distortion of the sphere axes; the radius isn't constant at 450mT. Rotation of the axes is not observable and will not be taken into account.

Afterwards, an algorithm by Yuri Petrov was used to perform a least-squares approximation and fit an ellipsoid onto the data set (Petrov, 2009). The algorithm was commanded to return the parameters of an ellipsoid offset from the origin but with axes aligned with the primary ones. The returned parameters where the offset vector and the ellipsoid axes lengths. This information was used to compensate the measurements and transform the data set back to a sphere centered on the origin. The formulas used are below:

$$m_x = \left(m_x - e_x\right)\frac{\|\mathbf{m}_i\|}{a_x}$$

$$m_y = \left(m_y - e_y\right)\frac{\|\mathbf{m}_i\|}{a_y}$$

$$m_z = \left(m_z - e_z\right)\frac{\|\mathbf{m}_i\|}{a_z}$$

Where **m** is the corrected magnetic field vector

m is the measured magnetic field vector

e is the offset vector

a is the vector containing the lengths of the ellipsoid axes

$\|\mathbf{m}_i\|$ is the known magnitude of the local magnetic field

The resulting locus is pictured below:

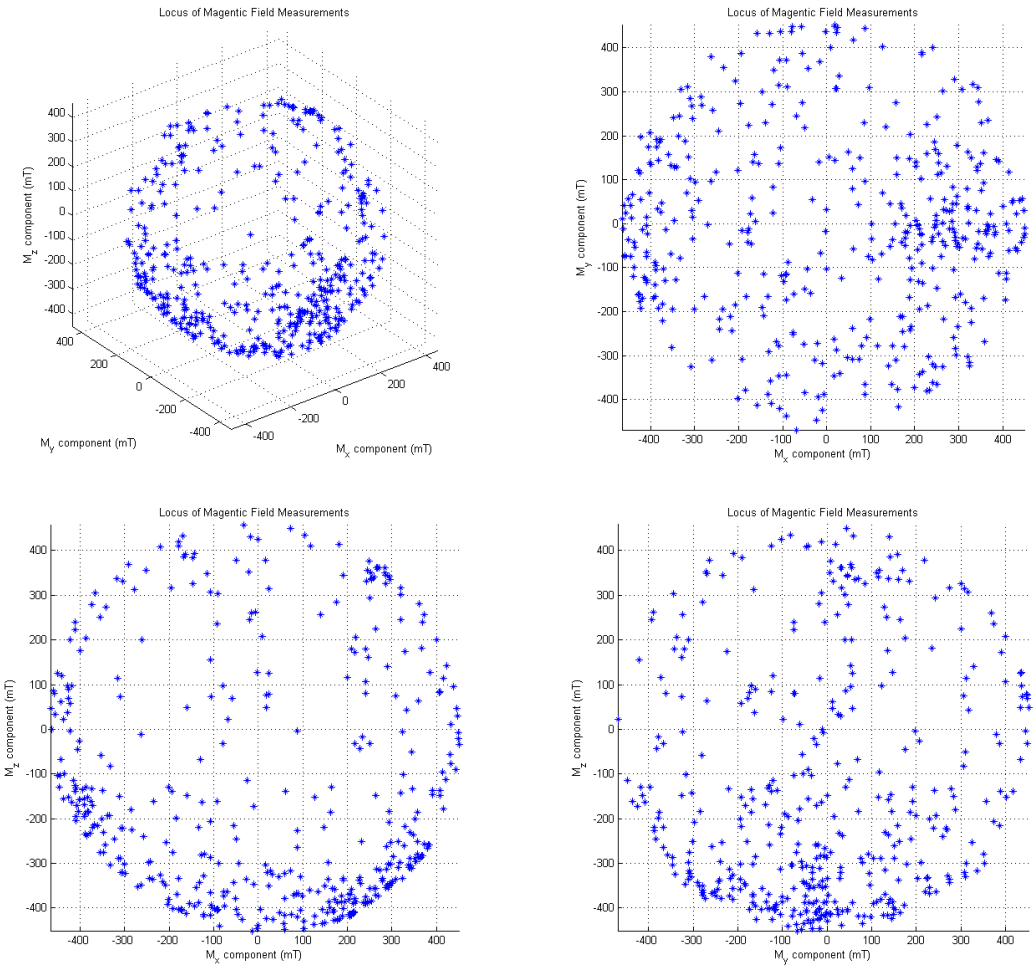


Figure 28: Compensated Magnetic Measurements

The procedures described above return a valid and dependable data set, which allows the AHRS algorithm to perform as accurately as possible.

It should be noted that this inverting transformation is valid only for that specific spatial hardware configuration of the module and the calibration procedure should be repeated for any new configuration.

Processing Unit

In the heart of the system lies the microcontroller board. This unit interfaces with the sensors to collect their data, runs the Motion Controller and Navigation Controller algorithms and produces the necessary outputs to drive the airframe motors.

Size and weight should be taken into account. The footprint of a full-fledged computer is too large to fit in light model airplanes.

Processing power is another consideration. The processor should be able to keep up with the necessary calculations that are needed in a very specific time frame. However, the typical control loop time in fixed-wing aircraft for non-demanding applications doesn't need to be very small, due to the inherent aerodynamic stability of the frame. A figure in the range of tens of milliseconds in enough to ensure stability and responsiveness.

Sufficient input/output ports should be provided by the microcontroller. As listed above, the sensors used require an array of Serial, Analog and I2C ports for communication. The servomotors and ESC of the aircraft require PWM signals to be driven.

Another factor is the programming environment of the microcontroller and its ease of use. Fast and robust IDEs as well as the existence of peripheral libraries and community support render the developing process much easier, letting the developer focus on the implementation of the algorithms. However, this comes at the expense of low-level processor control and accurate manipulation of the executed operations. In the end, the decision is usually a matter of preference and habit of the designer.

Popular platforms for amateur robotics are the Arduino, PIC, AVR and Raspberry Pie.

The Arduino platform was chosen in the end, due to its low cost, high-level programming, large support community and familiarity of the writer with the system. Moreover, Arduino is a popular platform for contemporary electronics and amateur robotics projects and this project showcases the capabilities of this ground-breaking piece of hardware.

For the needs of communication between the peripherals and the processor board, two separate serial ports were needed simultaneously. These cannot be found in the Arduino Mini or Uno, the lower-capability models. For that reason, the Arduino MEGA board was selected.

Specifically, the Arduino MEGA 2560 Rev3 board was selected. It is able to drive directly PWM controlled servomotors and interface with analog, PWM, I2C and serial communication peripherals. It is programmed directly via a USB port from a computer without requiring any change in the system hardware configuration. It can be powered with a DC voltage supply ranging from 5 to 12 volts and has very low power consumption.

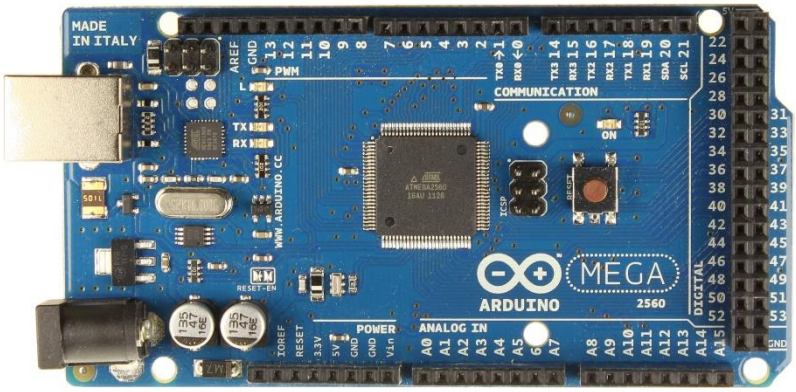


Figure 29: The Arduino Mega 2560

Data Logging

A black box is an essential part of any aerial platform and for good reason; while the system is in development phase, constant evaluation of its performance needs to be done. But since the vehicle flies hundreds of meters away from the operator, visual evaluation is impossible. This is where a data logger comes into play: It is a system where all system state variables, inputs and outputs are recorded regularly. With this data available after the mission end, the mission can be simulated and replayed in the computer and evaluated in detail, without time restrictions.

The hardware selected for this task was the openLog module by Sparkfun. It is a device that communicates via a Serial port. Any character written in its serial port is automatically recorded in the mounted microSD card. The SD card can then be removed and inserted in a computer for data extraction and analysis.

Figure 30: The openLog

In every software loop the following data is written in a Comma Separated Values file (.csv), which can then be parsed later easily by most stock or custom software.

1. Euler Angles
2. Manual control inputs (aileron, elevator, rudder, throttle)
3. Acceleration in 3 axis
4. Angular speed in 3 axis
5. Magnetic field in 3 axis
6. Manual override command
7. Flight mode selection switch input (mode 1 or mode 2)
8. Boolean value, corresponding to whether any manual input was commanded or not
9. Desired heading
10. Distance from next waypoint
11. Control input signals produced by the autopilot
12. Barometric altitude
13. Temperature recorded in the barometer
14. Barometric pressure
15. Pitot tube differential pressure
16. Airspeed
17. GPS horizontal dilution of precision
18. GPS coordinates
19. GPS fix age
20. GPS altitude
21. GPS course
22. GPS ground speed
23. GPS date and time
24. Software loop time

Airframe

Last but not least, the aerial platform should be mentioned. Even though it is not part of the autopilot module, it played a vital role in the development process.

In order to have a robust testing platform, able to provide adequate payload volume and weight, a 1.47m wingspan, EPO foam, electric RC airplane was chosen. It is in high-wing configuration, in order to facilitate the motion control process. Typical flight time for this type of technology is 7-10 minutes.

The make and model of the airframe is an ST Model Discovery.

Figure 31: The ST Model Discovery

5 Technical Difficulties/Considerations

5.1 Control module mounting

The 3-axes sensors inherently have the attribute of orientation which should match the orientation of the body frame. Thus, there is the need for accurate and stable mounting of the module on the aircraft frame. Rotation of the module mid-air would mean that the controller would have false measurements of the directional quantities. That would be catastrophic.

A mounting base was built out of foam board for the module, which fitted on the inside of the fuselage. Moreover, since vibration was present which could dislocate it, it was screwed on the fuselage.

As a final measure, during the initialization sequence of the system, the offset measurements are recorded and subtracted from all future measurements.

5.2 Telemetry

Initially, there was the intention of installing a two-way data link between the autopilot and a laptop ground station. That would serve both as a real-time telemetry link and an interactive mission control channel.

The hardware selected to implement that data link was a pair of radio modems that would simulate a Serial port over the air, between the two devices. Specifically, the Digi XBee 60mW 2.4GHz modules. Unfortunately, the writer was not aware at that time that this option would prove problematic: The aircraft radio control receiver, that must be constantly available for failsafe reasons, operates on the 2.4GHz channel as well. Having a 2.4GHz transmitter, the XBee, so close to a 2.4GHz receiver, even if they use different modulation methods, is a dangerous practice.

Even though the XBee module comes in a 900MHz version, its use in European countries is restricted by law; that radio band is licensed to cell phone communications.

Changing the hardware to overcome that problem would be an expensive procedure, both in money and time. Thus the data link option was abandoned.

5.3 Failsafe

The autopilot module built in the premises of this thesis is a prototype and has gone through relatively few hours of testing for each version. As a result, it cannot be deemed reliable and bug-free. A manual override control option must be available at all times to prevent a crash in case of software lock ups or hardware failures.

This is implemented by the addition of an 8-to-4 multiplexer IC and an extra microcontroller. The microcontroller reads the auxiliary channel of the radio control receiver which is linked to a switch in the radio transmitter. On the flip of that switch, the auxiliary microcontroller orders the multiplexer to route the control signals away from the Arduino, bypass it and send them unchanged directly to the servomotors and ESC; the model aircraft now flies completely manually, as if the autopilot isn't on it.

6 Results

6.1 PID gains

Controller	P-Gain	I-Gain	D-Gain	Lower Saturation Bound	Upper Saturation Bound	Sampling Time (ms)
Roll to Aileron	20	0.7	0.8	-201	185	65
Yaw to Roll	1	0.4	0.4	-30	30	65
Yaw to Rudder	10	0.1	0.1	-404	394	65
Pitch to Elevator	30	0.05	0.05	-399	568	65
Altitude to Pitch	0.04	0.01	0.01	-20	20	65
Airspeed to Throttle	60	0.5	0.1	-414	409	65
Airspeed to Pitch	4	0.02	0.01	-20	20	65

Table 1:PID Gains

For inner loop controllers, the system response is quite satisfactory. A fast transient response is achieved and the steady state error is small. An example is in the following figure, where the roll controller response is depicted, while it tries to reduce roll angle to zero.

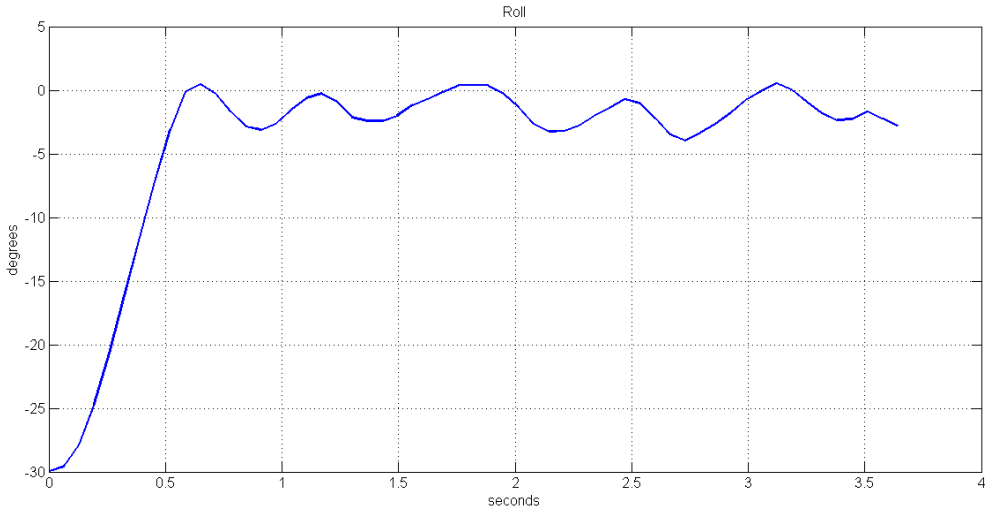


Figure 32: Roll-to-Aileron Controller Response

The same cannot be said for the outer loop controllers, where oscillations are present and manual tuning proved to be much more difficult. While the general level of the controlled variable is achieved, the steady state response isn't satisfactory. This is intensified even more by the coupling of the lateral and longitudinal axis.

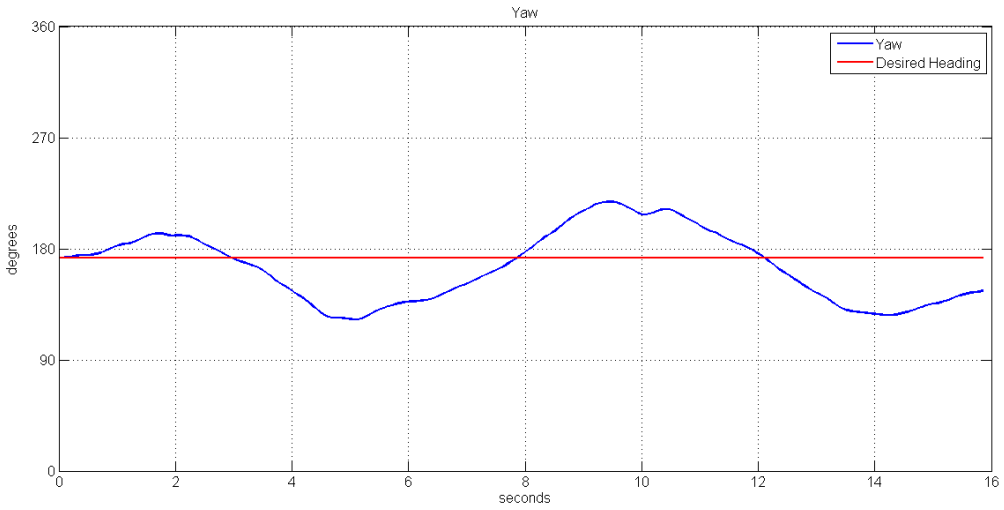


Figure 33: Yaw-to-Roll Controller Response

6.2 Typical flight mission overview

Below is presented a typical flight mission, in the form of map traces and graphs. The results were extracted from the in-flight datalogs, processed in Matlab and plotted on Google Earth with the help of the KML Toolbox for Matlab, by Rafael Oliveira (Oliveira, 2013)

Take-Off

The threshold airspeed for take-off was is at 10m/s. After the target airspeed is reached, the climb angle is set at 20 degrees. The take-off procedure is supposed to end at 30 meters, where the autopilot must level the plane and engage the first waypoint.

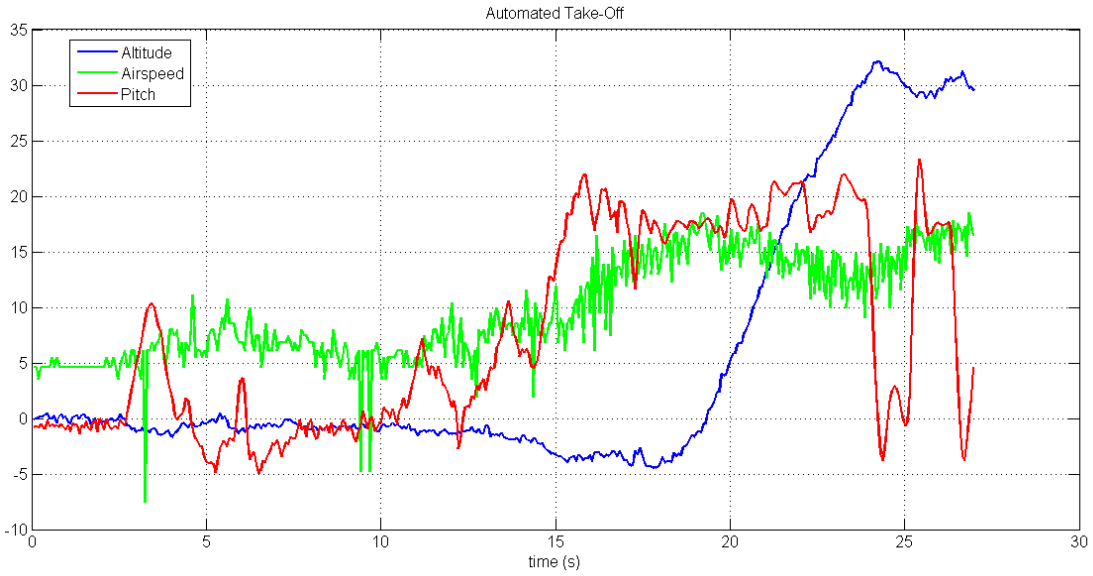


Figure 34: Take-Off System Response

Waypoint Following

Next, a few waypoints are linked sequentially to form a flight mission. Unfortunately, due to the slow response of the yaw-to-roll controller, the turns are not fast and crisp. An overshoot in bot turns is observed. The is first from the path start (lower right) towards the first waypoint and the second is after the first waypoint is cleared and the plane turns West towards the second waypoint.

Generally, altitude is followed quite effectively during the mission.

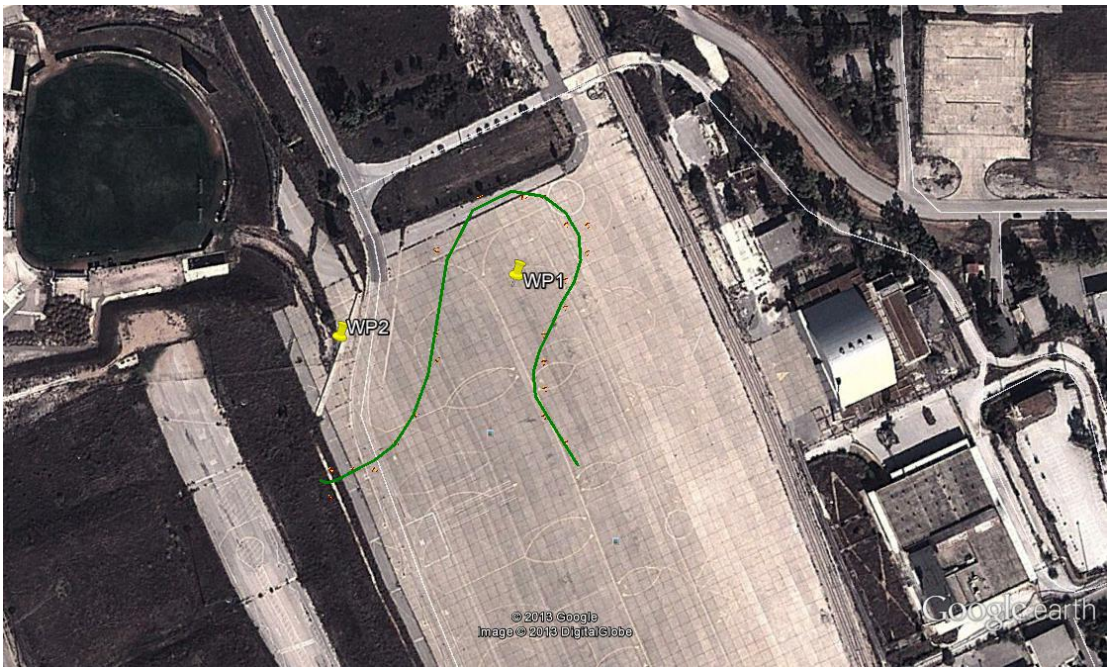


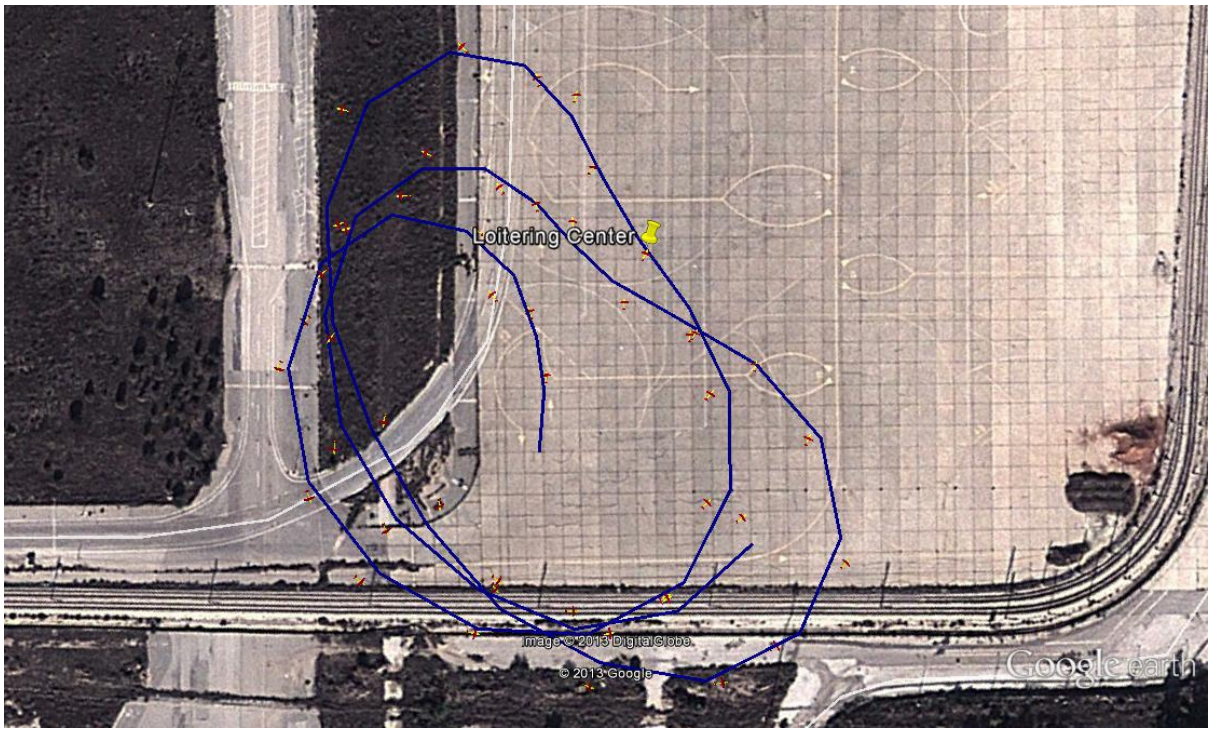
Figure 35: Waypoint Visiting

Loitering

Finally, the aircraft was programmed to loiter around a specific set of coordinates. The top-down view of the loiter pattern is shown below. The loitering center is depicted as a yellow pin on the map. In the second picture, the vector field direction over each point of the flight path is visible.

Notice that the circling pattern isn't centered around the pin. This is accredited to the North wind blowing steadily in the area, along with the sluggish response of the Motion Controller. Combined, these two factors force the airframe to turn West prematurely, while it is pointed to the North direction, thus undershooting the circular pattern.

The altitude while performing the maneuver varies, because of the sudden change of lift, as the wind direction changes. In the long run, however, the autopilot manages to hold the desired altitude.



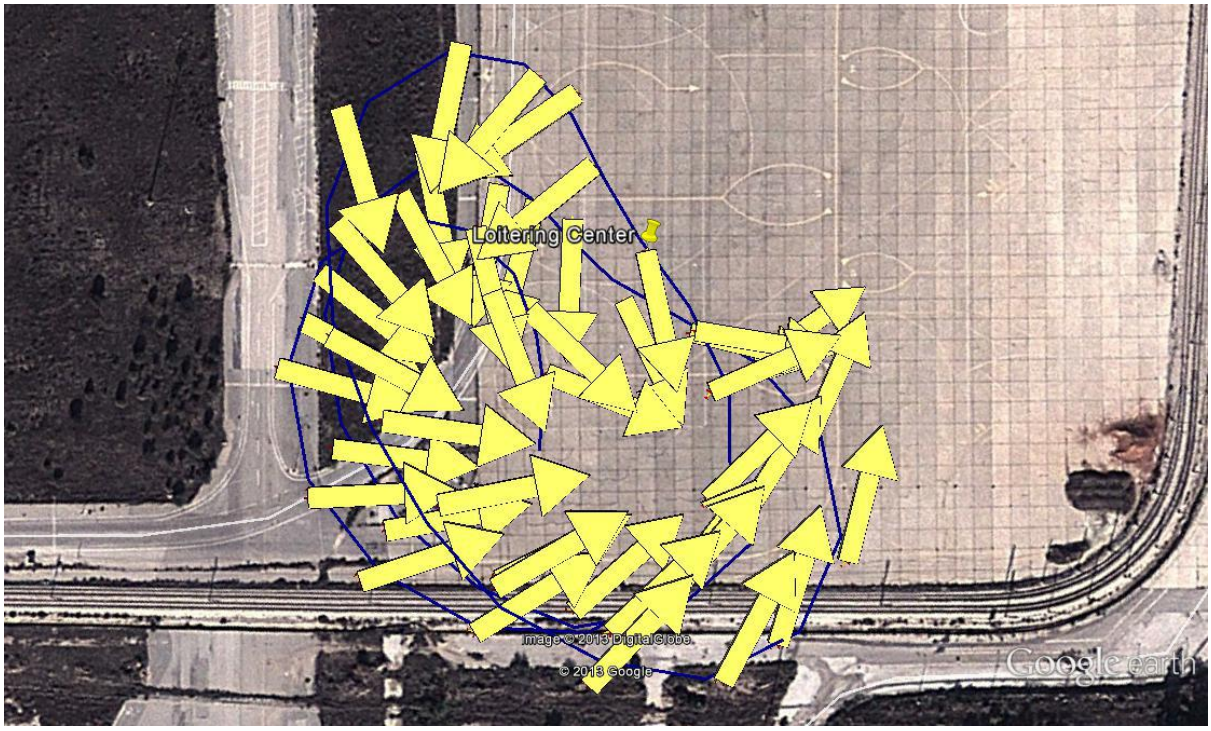


Figure 36: Loiterning Depiction on Map

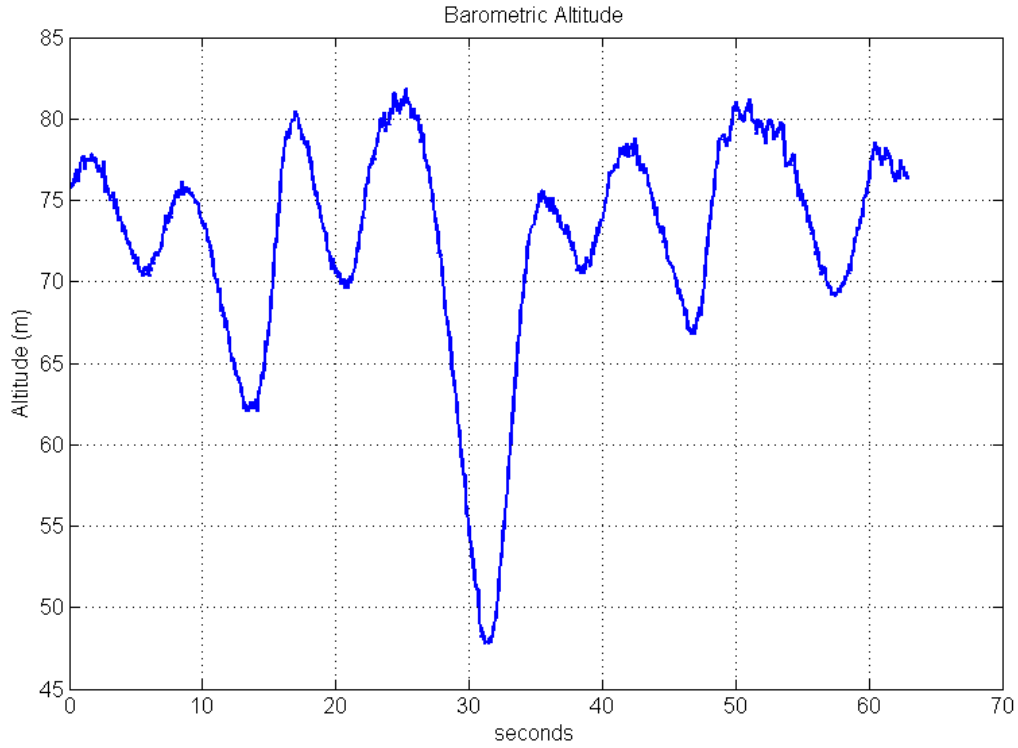


Figure 37: Altitude During Loiter

6.3 Power budget

Part	Voltage (V)	Current (mA)	Power (mW)
Arduino Mega 2560	5.0	60	300
Picaxe 08M2	5.0	0.1	0.5
MPU6050	3.3	3.7	13
HMC5883L	3.3	0.1	0.4
UP501	3.3	-	75
BMP085	3.3	0.2	0.7
MPXV7002DP	5.0	10	50
openLog	5.0	6	30
Total	-	-	**470**

Table 2: Power Budget

6.4 Module Cost

The cost is calculated based on the most costly parts. Cheap, miscellaneous parts are omitted.

Part	Price (€)
Arduino Mega 2560	39.00
MPU6050	30.53
HMC5883L	11.45
UP501	30.53
BMP085	19.08
Airspeed Measurement Kit	22.90
openLog	19.08
Manufacturing/Structural	30.53
Total	**203**

Table 3: Bill of Materials

7 Issues of further research

- A more sophisticated Navigation Controller: At this point, the Navigation Controller is nothing more than a compass pointing the aircraft to the desired heading at the right time. This is the most basic form of a Navigation Controller and still has a lot of room for improvement.
  The first step would be to set up a dynamic vector field to enable following the line that links two consecutive waypoints, instead of simply targeting the next waypoint. This is important, since it provides more functionality to the waypoints, which now can be used to survey a predefined surface area.
  Next, a more intelligent way to label a waypoint visited could be sought. Disturbances can force the UAV to overtake a waypoint sideways and if the labeling criterion is distance from waypoint, then the UAV may turn back to reach the waypoint. This isn't the required behaviour. Instead, a waypoint could be labeled as visited if the UAV is past the line perpendicular to the one linking the previous and the next waypoint, which goes over the current waypoint.
  Finally, a better way to negotiate the turns would be useful. Anticipating the waypoint visit and starting the turn early, placing the waypoint at its apex shortens the flight distance and saves flight time.

- Fine tuning the motion controller: As was discussed in Chapter 5, completely satisfying gains for the Motion Controller were not found. Finding a method to produce PID gains in absence of a mathematical model for the aircraft, based on available sensory data should greatly improve the capabilities of the system.

- Developing a landing mode for the navigation controller, with the aid of an ultrasonic sensor: For a system to achieve complete autonomy, it must be able to complete its mission and return to the ground safely. This is not possible with the current array of sensors, which cannot detect the ground and the distance of the UAV from it.
  An ultrasonic sensor pointing to the ground could make an effective and cheap solution to the problem, giving the UAV the ability to perform soft touch-downs and also avoid unexpected obstacles.

- Allowing for payload handling of the system: For the UAS to have true usefulness, some kind of payload should be mounted on the airframe, whether be it a simple supply drop hatch or an advanced gimbaled video camera. In any

case, the UAS onboard computer should be able to handle those external modules with dedicated signal ports. However, this is system-specific and will only be sought should the need arise.

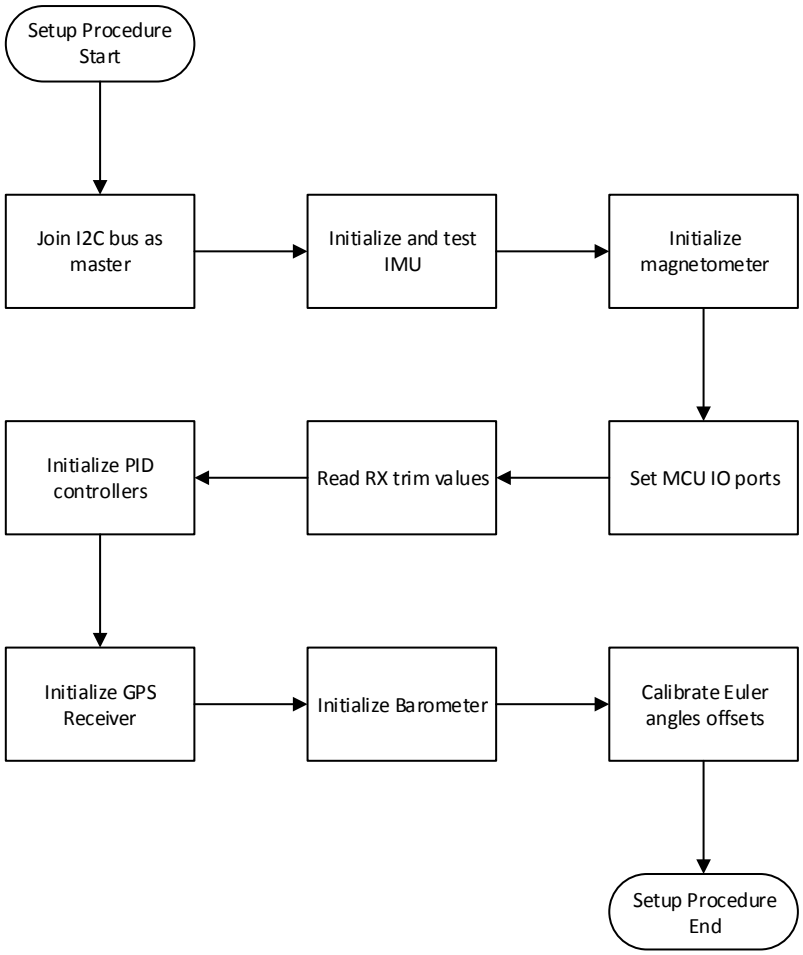
- Integrate FPV equipment: Installing FPV equipment in the airframe is a great and simple way to add interactivity, functionality and marketing value. It can also be combined with automated surveillance missions, providing a live video downlink.

- Integrate a more user friendly flight plan creator: At this point, the flight plan and the waypoint list is nothing more than lines of code in the firmware. Each time a new flight plan is needed, the waypoints are re-coded in the firmware and uploaded on the Arduino. While this is relatively simple, it severely extends time-to-take-off and looks unprofessional. Programming a Graphic User Interface from scratch is barely productive. Instead, integration with an existing software is much more lucrative, such as MAVLINK, which also provides a lot more features.

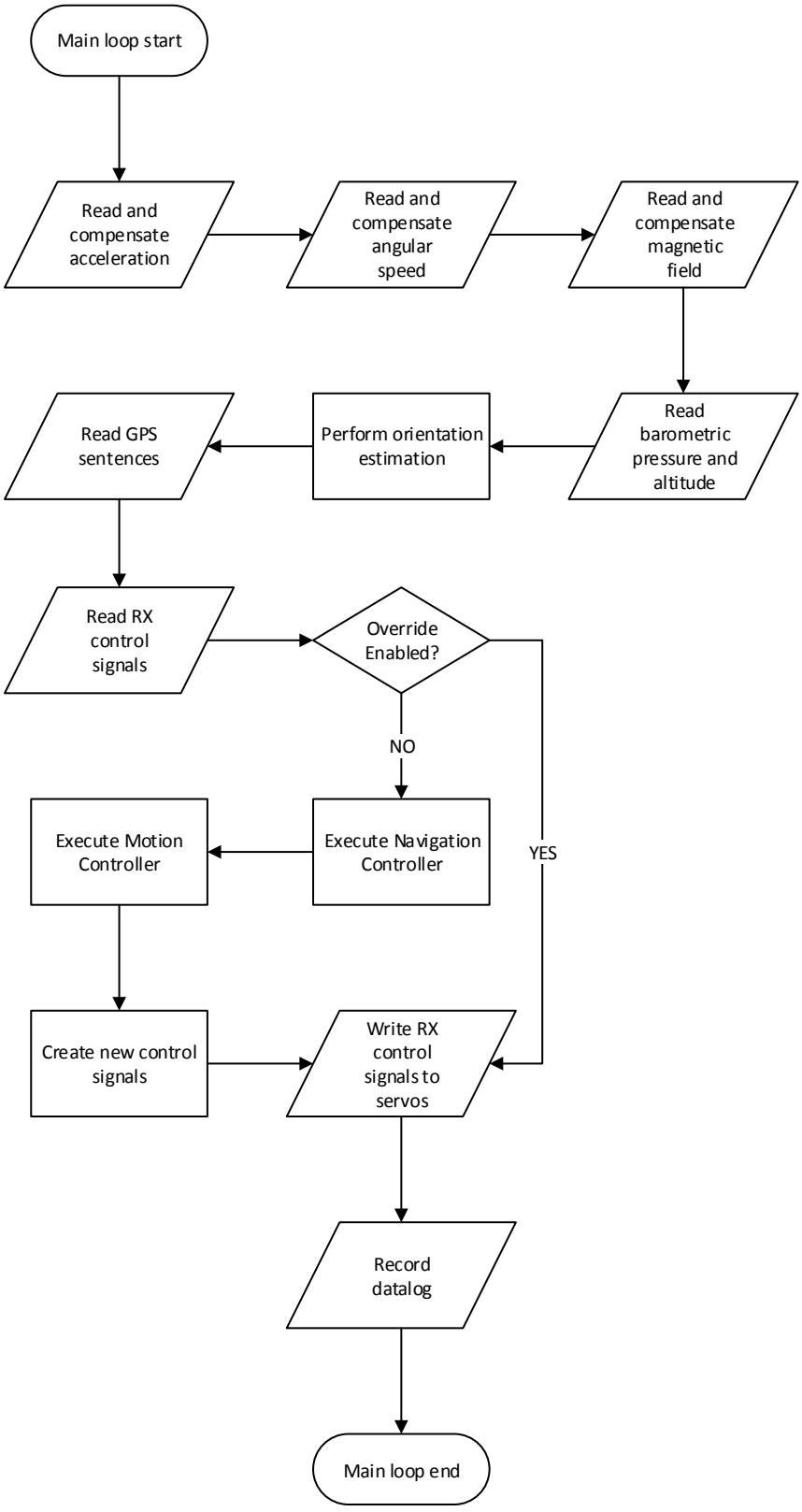
- Investigate the performance of the module in other airframes: One of the initial goals of this system was for it to be interchangeable between different airframes. Whether it can truly operate similarly on different aircraft without much compromise in performance remains to be tested.

Appendix A: Software Logic Diagrams

Setup procedure



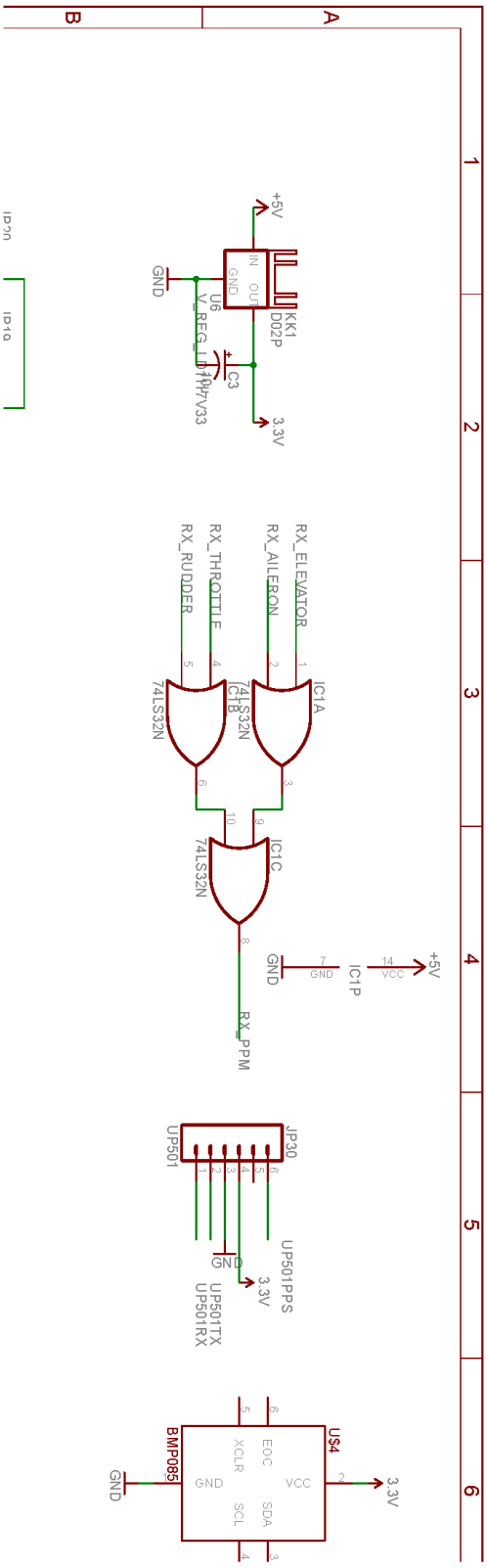
Main Loop

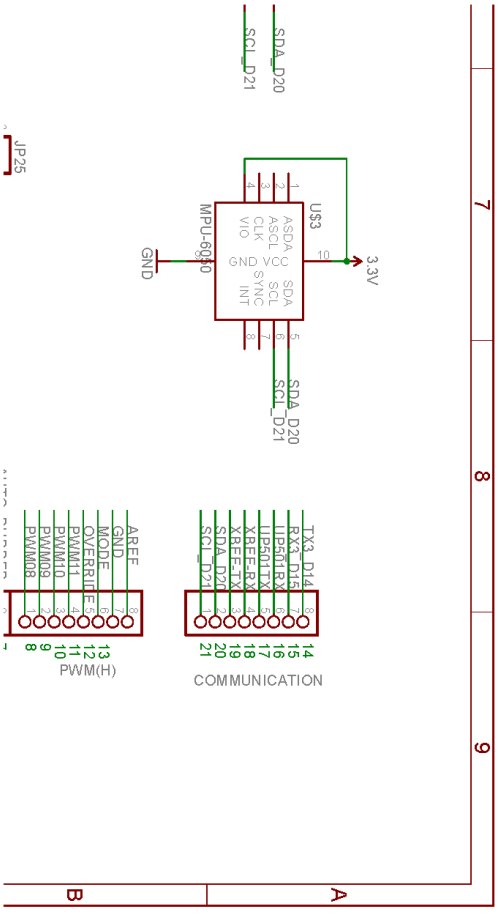

Main loop start
Read and compensate acceleration
Read and compensate angular speed
Read and compensate magnetic field
Read barometric pressure and altitude
Perform orientation estimation
Read GPS sentences
Read RX control signals
Override Enabled?
NO
YES
Execute Navigation Controller
Execute Motion Controller
Create new control signals
Write RX control signals to servos
Record datalog
Main loop end

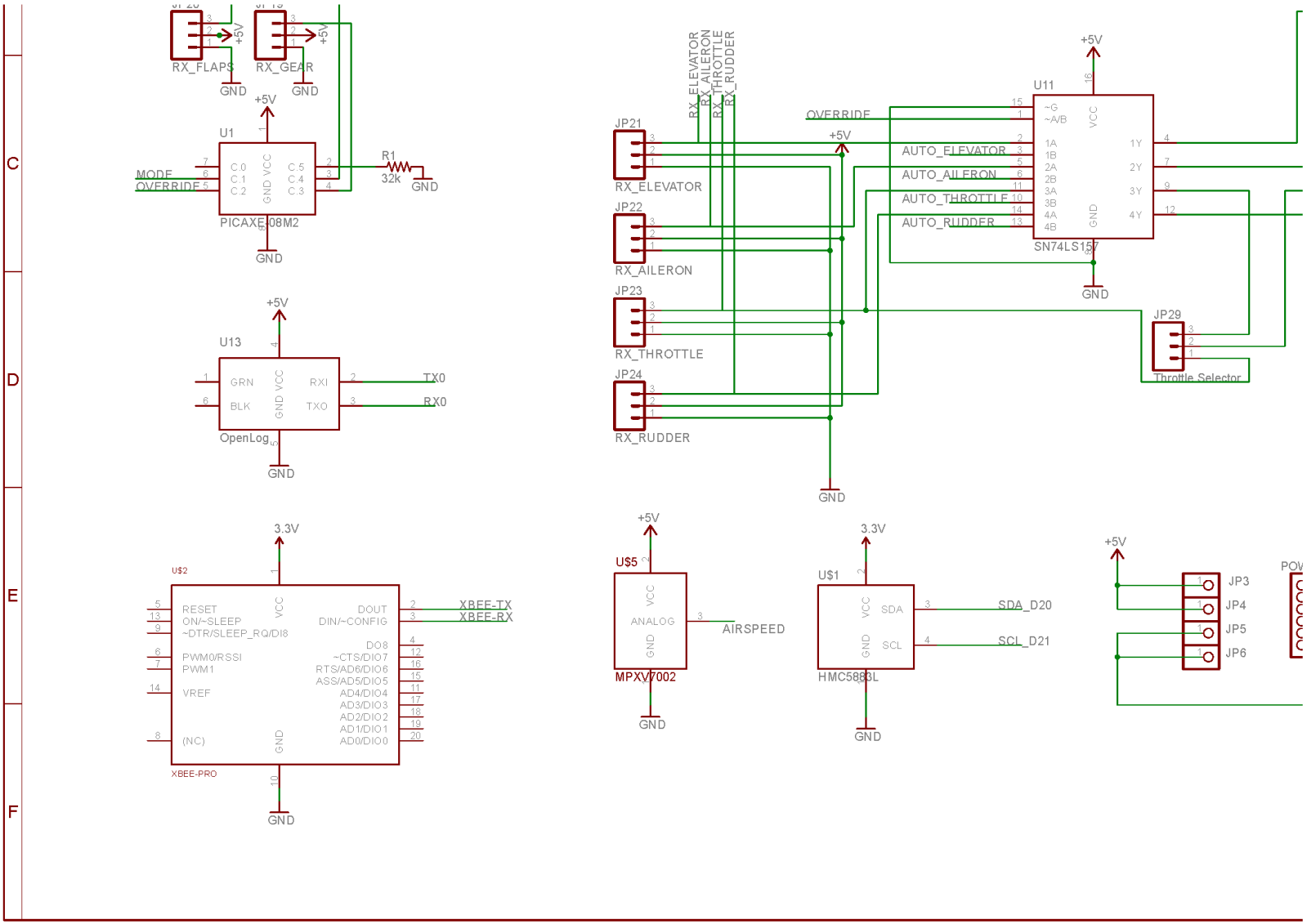
Appendix B: Failsafe Controller Firmware

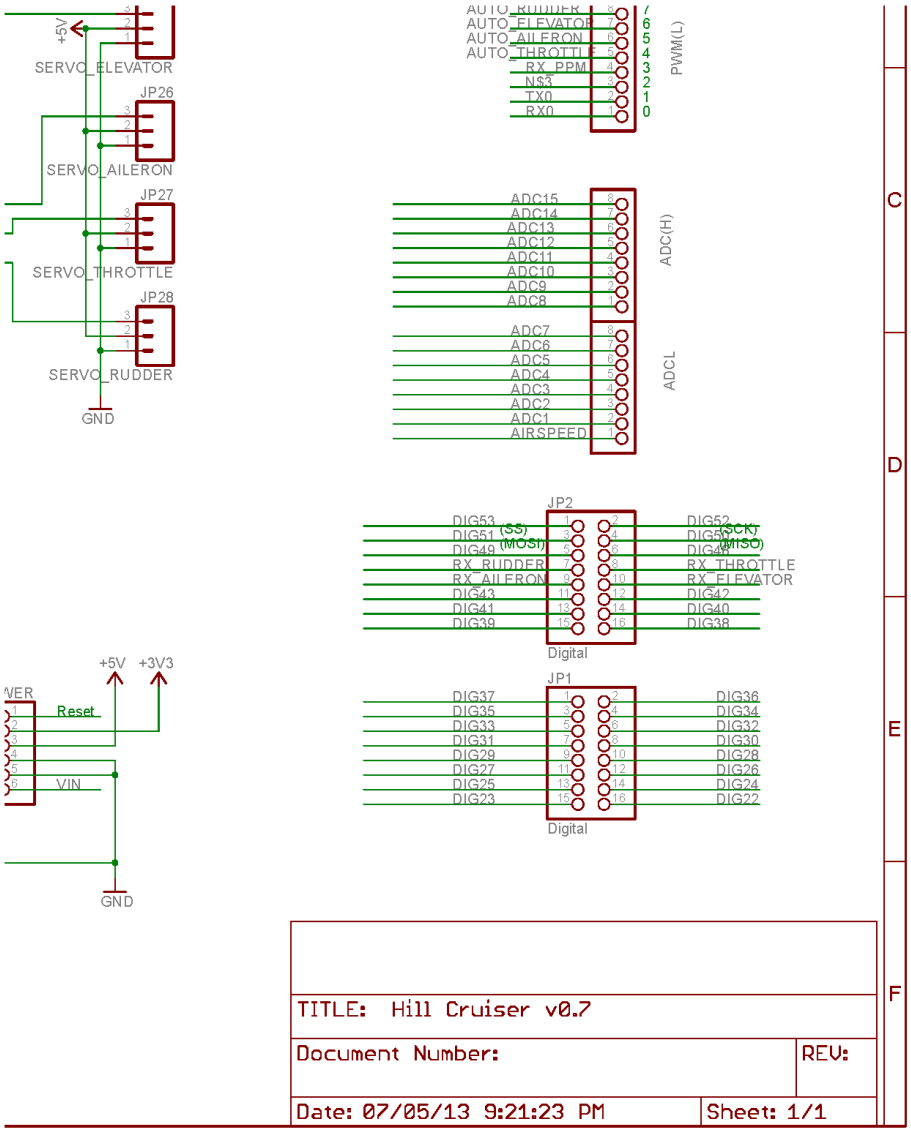
```
main:
     input C.3
     input C.4
     output C.2
     output C.1
ret: pulsin C.3,1,w0
     if w0<=180 then
          low C.2
     else
          high C.2
     endif
     pulsin C.4,1,w0
     if w0<=180 then
          low C.1
     else
          high C.1
     endif
     goto ret
```

Appendix C: Electrical wiring diagram




SCL_D21
SDA_D20
JP25
MPU-6050
US3
GND
3.3V
SDA_D20
SCL_D21
AREF
GND
MODE
OVERRIDE
PWM11
PWM10
PWM09
PWM08
TX3_D14
RX3_D15
UP501RX
UP501TX
XBEE-RX
XBEE-TX
SDA_D20
SCL_D21
PWM(H)
COMMUNICATION
7
8
9
A
B


RX_FLAPS
RX_GEAR
GND
+5V
U1
MODE
OVERRIDE
PICAXE-08M2
R1
32k
U13
OpenLog
GRN
BLK
RXI
TXO
TX0
RX0
U$2
XBEE-PRO
RESET
ON/~SLEEP
~DTR/SLEEP_RQ/DI8
PWM0/RSSI
PWM1
VREF
(NC)
DOUT
DIN/~CONFIG
XBEE-TX
XBEE-RX
3.3V
RX_ELEVATOR
RX_AILERON
RX_THROTTLE
RX_RUDDER
JP21
JP22
JP23
JP24
OVERRIDE
AUTO_ELEVATOR
AUTO_AILERON
AUTO_THROTTLE
AUTO_RUDDER
U11
SN74LS157
JP29
Throttle Selector
U$5
MPXV7002
ANALOG
AIRSPEED
U$1
HMC5883L
SDA
SCL
SDA_D20
SCL_D21
JP3
JP4
JP5
JP6
A
B
C
D
E
F


+5V
SERVO_ELEVATOR
JP26
SERVO_AILERON
JP27
SERVO_THROTTLE
JP28
SERVO_RUDDER
GND
AUTO_RUDDER
AUTO_ELEVATOR
AUTO_AILERON
AUTO_THROTTLE
RX_PPM
TX0
RX0
PWM(L)
ADC15
ADC14
ADC13
ADC12
ADC11
ADC10
ADC9
ADC8
ADC(H)
ADC7
ADC6
ADC5
ADC4
ADC3
ADC2
ADC1
AIRSPEED
ADC(L)
JP2
DIG53
DIG51 (SS)
DIG49 (MOSI)
RX_RUDDER
RX_AILERON
DIG43
DIG41
DIG39
DIG52 (SCK)
DIG50 (MISO)
RX_THROTTLE
RX_ELEVATOR
DIG42
DIG40
DIG38
Digital
JP1
DIG37
DIG35
DIG33
DIG31
DIG29
DIG27
DIG25
DIG23
DIG36
DIG34
DIG32
DIG30
DIG28
DIG26
DIG24
DIG22
Digital
+5V
+3V3
Reset
VIN
GND
TITLE: Hill Cruiser v0.7
Document Number:
REV:
Date: 07/05/13 9:21:23 PM
Sheet: 1/1

Appendix D: Installation pictures

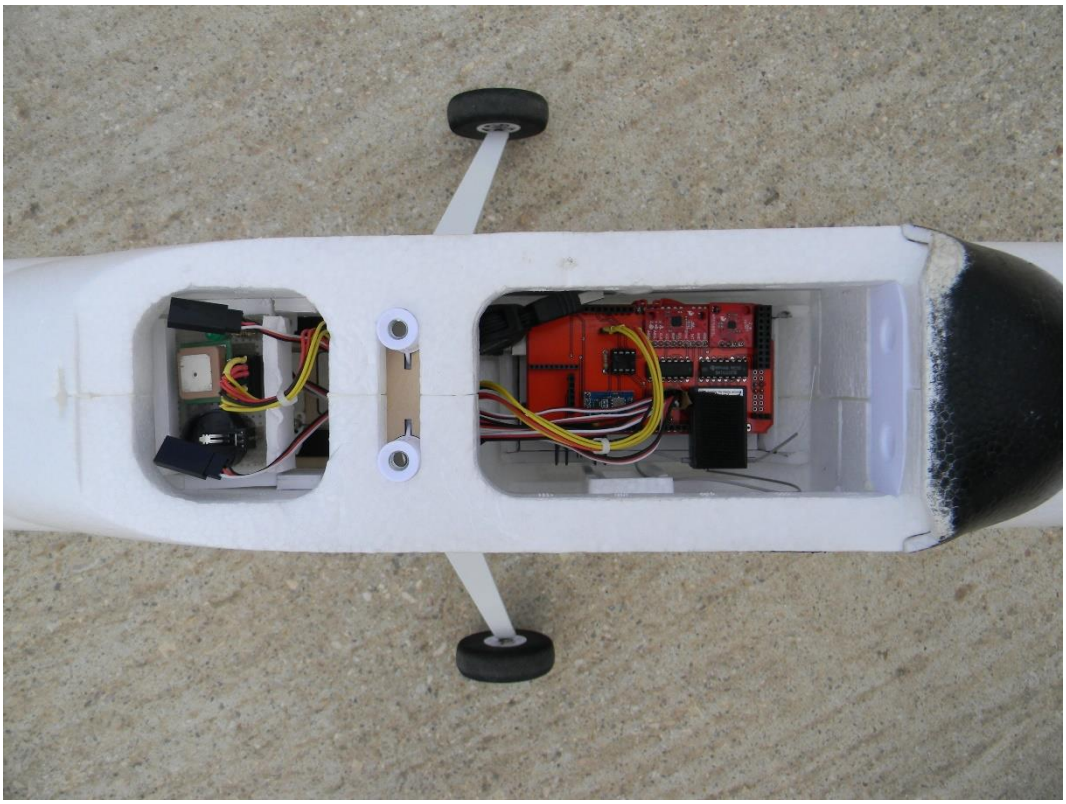


Figure 38: Top View of the Autopilot Installation

Figure 39: Front View of the Airframe

References

Beard, R., & McLain, T. (2012). *Small Unmanned Aircraft.* Princeton.

European Space Agency. (2013). *egnos navigation*. Retrieved from ESA: http://www.esa.int/Our_Activities/Navigation/The_present_-_EGNOS/What_is_EGNOS

Fastrax Ltd. (2010, August). *NMEA Manual for Fastrax IT500.* Retrieved from http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/GPS/NMEA%20manual%20for%20Fastrax%20IT500%20Series%20GPS%20receivers_V1.7.pdf

Freescale. (2012, February). *Layout Recommendations for PCBs Using a Magnetometer Sensor.* Retrieved from freescale.com: http://cache.freescale.com/files/sensors/doc/app_note/AN4246.pdf

Freescale. (2013, April). *Calibrating an eCompass in the Presence of Hard and Soft-Iron Interference.* Retrieved from freescale.com: http://cache.freescale.com/files/sensors/doc/app_note/AN4246.pdf

Graham, M. (2011, July). *GPS versus barometric altitude: the definitive answer*. Retrieved from Cross Country: http://www.xcmag.com/2011/07/gps-versus-barometric-altitude-the-definitive-answer/

Houghton, E., & Carpenter, P. (2003). Aerodynamics for Enginerring Students. Butterworth-Heinemann.

Madgwick, S., Harrison, A., & Vaidyanathan, R. (2011). Estimation of IMU and MARG orientation using a gradient descent algorithm. *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on.*

Movable Type Ltd. (n.d.). *Calculate distance, bearing and more between Latitude/Longitude points*. Retrieved from http://www.movable-type.co.uk/scripts/latlong.html

NOAA, National Geophysical Data Center. (2013). *Geomagnetism*. Retrieved from ngdc.noaa.gov: http://www.ngdc.noaa.gov/geomag/geomag.shtml

Oliveira, R. (2013). *KLM Toolbox v2.6.* Retrieved from Matlab Central: http://www.mathworks.com/matlabcentral/fileexchange/34694-kml-toolbox-v2-3

Petrov, Y. (2009). *Ellipsoid Fit.* Retrieved from Matlab Central: http://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit

Weisstein, E. W. (n.d.). *Great Circle*. Retrieved from MathWorld: http://mathworld.wolfram.com/GreatCircle.html

Wikipedia contributors. (2013, April). *Barometric formula*. Retrieved from Wikipedia, The Free Encyclopedia: http://en.wikipedia.org/wiki/Barometric_formula

World Data Center for Geomagnetism. (2010). *International Geomagnetic Reference Field (IGRF-11).* Retrieved from http://wdc.kugi.kyoto-u.ac.jp: http://wdc.kugi.kyoto-u.ac.jp/igrf/