ABSTRACT

This project focuses on the implementation of an inexpensive autonomous Unmanned Aerial Vehicle (UAV) system. By its nature, it requires two fundamental elements to operate, the aerial vehicle itself and a Ground Control Station (GCS) software.

For the aerial vehicle, a quad rotor helicopter is built based on a design named Aeroquad [1]*. It is a remote controlled quad copter with an embedded microcontroller and several sensors. Its software is open source making it cheap and flexible to modify. For the completion of this project, the open source software is integrated to a UAV featuring:*

- *Autonomous navigation between different positions defined by the user*
- *Communication link establishment with the GCS using an Internet connection*
- *Uninterruptable flight on communication link error events*
- *Amendment of missions during flight*

On the other hand, GCS software is developed for the control and configuration of the UAV. This software successfully provides the user options to:

- *Calibrate the vehicle sensors*
- *Configure the communication link*
- *Plan the UAV's mission,*
- *Track the vehicle on the globe in real time*
- *View the current state of the UAV's orientation*

NOMENCLATURE

ADC	Analogue to Digital Converter
DCM	Direction Cosine Matrix
DOF	Degrees Of Freedom
ESC	Electronic Speed Controller
GCS	Ground Control Station
GIS	Geographic Information System
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile
HUD	Head-Up Display
I^2C	Inter-Integrated Circuit
IMU	Inertial Measurement Unit
MEMS	Micro Electro Mechanical Systems
NMEA	National Marine Electronics Association
PCB	Printed Circuit Board
PID	Proportional Integral Derivative
PWM	Pulse Width Modulation
RC	Remote Controlled
TCP/IP	Transmission Control Protocol / Internet Protocol
UAV	Unmanned Aerial Vehicle
UCS	Universal Character Set
UART	Universal Asynchronous Receive Transmit
UTF-8	UCS Transformation Format — 8-bit
VTOL	Vertical Take Off and Landing
WGS84	World Geodetic System (revised in 2004)

ACKNOWLEDGEMENTS

I would like to thank my project advisor Dr. Paul Warr for his valuable advice and guidance and inspiration during my studies at University of Bristol and specifically for the advice towards the completion of this project. Also, I would like to thank all of the academic staff for the knowledge I earned during this year. I would also like to thank my friend Yiannis helping me with proof reading. Last but not least, I would like to express my gratitude to my family for their support during my studies.

TABLE OF CONTENTS

List of Figures

List of tables

1. INTRODUCTION

UAVs were firstly and still being developed for military purposes such as covert reconnaissance, security and post disaster analysis. Over the past decade, UAVs demand have been grown exponentially when such a system stopped serving just military, but also civilian needs. They are nowadays found in a variety of fields. Geographic Information System (GIS) services are using UAVs for maps updating. They are also being used by environmental analysts for various observations and predictions. Some movie scenes were also taken by such vehicles.

What it makes these systems more and more popular, is the reduced complexity of their flight control which can be achieved by a non-expert pilot. All the difficulties of controlling commands are now done by software enabling a user to fly them easily and securely. Furthermore, vertical take-off and landing (VTOL) UAVs have a big advantage over the other types as they are commonly used in areas consisting of abrupt altitude changes such as inhabitant areas. Hence, rapid vertical climb or vertical decent is crucial. This project focuses on the implementation of a complete UAV system. It includes both the aerial vehicle software and hardware needs as well as the GCS platform.

For the aerial vehicle, a low-cost remote controlled quad-rotor helicopter was used as it features stable flights [1]. The control software is open source and the computational unit is a microcontroller which can be programmed to meet the flight autonomy requirements. This also reduces its cost, which is a main constraint of the project.

Within this project, a low cost remote controlled model is transformed into an efficient UAV able to navigate autonomously. Furthermore, the aircraft is able to navigate safely to its station in case of a communication link error. The communication between the aircraft and the ground control station is long ranged as the General Packet Radio Service (GPRS) was used.

Consequently, a GCS application was developed. This application is responsible to initialize the aircraft by performing several pre-flight tasks. It also provides a user friendly environment for mission planning and is programmed to receive real-time information that is coming from the UAV. Last but not least, it visually tracks the aircraft's route on map while keep logs of the flight data for further analysis or simulation.

For the development of this project several areas were studied. These areas cover both software and hardware implementation needs. Sensor interfaces, communication link protocols and different technology availabilities were studied. Furthermore, mathematical models regarding the aircraft orientation have been analysed. Finally, the components should exist on a state-of-the-art ground control system application and the role of each was discovered and implemented in a way providing a user friendly interface.

1.1. Aim and Objectives

Although UAVs are very popular, they are not found in civilian applications yet due to their high cost on research and electronic components required for their development.

The aim of the project is the integration of an inexpensive, open source, remote controlled quad rotor helicopter to an efficient UAV. This includes the design and the implementation of the aircraft's frame, the Printed Circuit Board (PCB) design interfacing the required sensors, as well as the integration of the current software to an autopilot.

The second main part of the project is focused on the implementation of Ground Control Station (GCS) software able to interact in real time with the aerial vehicle. Furthermore for the communication of the two systems, an application protocol has been defined that makes use of the Transport Control Protocol/Internet Protocol (TCP/IP) making the system web based.

1.1.1 Unmanned Aerial Vehicle Objectives

- Design of the aircraft's frame
- Design of a PCB interfacing the required sensors with a microcontroller
- Integration of the software able to receive and transmit data in real time using General Packet Radio Service (GPRS)
- Integration of the software to hold its altitude set by the user during flight
- Integration of the software to hold it position at a waypoint
- Integration of the software to travel between two waypoints
- Integration of the software to navigate securely on communication link failures
- Integration of the software to perform auto take-off and auto landing

1.1.1.1 UAV Objectives Analysis

Design of the aircraft's frame:

The frame of the UAV is designed by following the guidelines of the open source quad rotor helicopter [1]. Its setup is selected to be medium sized as it offers stability over wind, further payload capabilities and long flights.

Design of a PCB interfacing sensors with the microcontroller:

The PCB design and implementation was a result of studying the sensors of the quad rotor's architecture [1]. It was then amended to a new design featuring all the required sensors to provide an autonomous flight such as:

- 3 axis gyroscope for attitude calculations
- 3 axis accelerometer for roll-pitch drift corrections
- 3 axis magnetometer for autonomous navigation complementing Global Positioning System's (GPS).
- GPS for autonomous navigation and yaw corrections
- GPRS for communication

Integration of the software able to receive and transmit data using GPRS service:

The communication link between the microcontroller and the GPRS module has been implemented. The system is able to manipulate sockets for data transmission and reception. Furthermore, an application layer protocol between the aircraft and the GCS has been defined. A sample of such a protocol is presented in the "Theoretical Background on GCS", found in chapter 3. The UAV is able to have a real time bidirectional communication link with the GCS over the Internet. The communication link is used for the real time amendment of the UAV's mission, as well as for the generation of the artificial instrumentation and tracking on the GCS.

Integration of the software able to hold its altitude set by user:

The aircraft makes use of the barometer to hold a user set altitude during navigation. As there is no low cost sensor to calculate altitude over ground, but both GPS and barometer measures altitude over sea level, it is left to the user to set a safe altitude for navigation.

Integration of the software able to hold its position:

Position hold is the ability of the aircraft to hover at the destination point by utilizing GPS. For the achievement of that objective an algorithm has been developed taking into account the inaccuracy of the positioning system as it is usually about a radius of 10-20 meters. The algorithm stabilises the aircraft regardless any external force (such as wind) applied on the aircraft. The aircraft is designed to hold its position on the last position that exists in the mission list. However, a user can use the GCS-UAV communication to set any position he or she decides as last position.

Integration of the software able to travel between two coordinate positions:

The UAV vehicle is able to travel between different coordinate points autonomously. The software implementing this feature is analysed in chapter 4.2.1.

Integration of the software to return to base on a communication link failure:

Communication links are vulnerable, especially when it is built on the unreliable Global System for Mobiles (GSM) service and the Internet where each packet of data has to travel through some active network equipment (switches, routers) until the destination is reached. For this reason the UAV is designed to accomplish its missions even in the case of a communication link failure. After the last position is reached (base), the UAV hovers holding its position until the operator takes the manual control to land it.

Integration of the software to perform auto take-off and landing:

The UAV should utilize the capabilities of the barometer and a range finder sensor to perform automatic take-offs and landings controlled by the GCS.

.

1.1.2 Ground Control Station Objectives

- Implementation of an application layer protocol between the UAV and the GCS.
- Design of a GCS modularized as observed in the GCS background chapter 3 containing:
  - Mission planning module
  - Flight data – instrumentation visualizer module
  - Configuration module

1.1.2.1 GCS Objectives Analysis

Design and implementation of a communication protocol between the UAV and the GCS

The TCP/IP transport protocol was used. On top of it, a telnet-type application layer protocol was defined implementing the communication between the UAV and the GCS. The application protocol is a similar message with the one covered by sub-chapter 3.3 in the "Theoretical background on GCS".

Design of a ground control station modularized as observed in the theoretical background chapter 3

The GCS software was developed in a visual object-oriented language. The software has mainly three views (modules), each one performing a role. These views are:

- Mission planning
- Flight Data
- Configuration

The GCS makes use of open source Google Earth's libraries for the virtual globe presentation. Asynchronous socket programming algorithms written for the communication link establishment and Serial Port programming were used for the wired communication with the vehicle for its configuration, initialization and calibration of pre-flight procedures.

Implementation of mission planning module:

The mission planning view is user friendly environment where a user can design the desired missions for the aircraft to travel. The GCS makes use of open source Google Earth's libraries for the virtual globe presentation. The positions are placed by the user as 'clicks', are then converted to missions and are sent to the UAV.

Implementation of flight data view of the UAV:

The GCS tracks the position of the aircraft in real time. This is done by analysing the data sent from the UAV to the GCS. The data is visualised on the map not just as a dot, but as a moving arrow clarifying the UAV's orientation with respect to North.

Implementation of artificial instrumentation of the UAV:

The user, despite the tracking information is able to observe the current state of the vehicle in terms of its orientation. It is provided with information such as: number of satellites the GPS is listening from, the speed over ground and inertial information like pitch and roll angles with respect to level position. Last but not least, the user can view the azimuth of the aircraft.

2. THEORETICAL BACKGROUND ON UAVS

Despite that the project can be characterised as an implementation project, several areas were studied for the implementation of each objective. This chapter is dedicated to the research made on the UAVs and especially the VTOL ones. It analyses the different approaches used to control the aircraft's orientation and stabilisation which are then used to develop the autopilot and the communication system.

2.1. Unmanned Aerial Vehicles (UAVs)

Unmanned Aerial Vehicles are characterised by small size and light weight [2]. They are also defined as unmanned (non-pilot) systems. Therefore, they make use of a wide variety of sensors to fill this absence in order to have stable and accurate flights. Due to their small size, multiple sensors and minimal space, they usually have some limitations, as most embedded systems do. These features and limitations in payload, size, power resources and computation power is what it needs to be balanced during the design of such a system.

Two main types of UAVs are available [3]; the rotary wing and fixed wing. Both of them have their advantages as well as some disadvantages. Fixed wing models feature less power consumption than rotary wing. On the other hand, they require enough distance for take-off and landing. Rotary wing models are also divided in two categories; the helicopters and multi-rotor helicopters (same axis – reverse rotation) with latter featuring simplified design but what's more, stable flights.

Each aircraft (fixed-wing or rotary-wing) has three flight dynamic parameters known as pitch, roll and yaw as shown in Figure 1. These are basically three angles which are used to describe the orientation of the UAV at any given time.

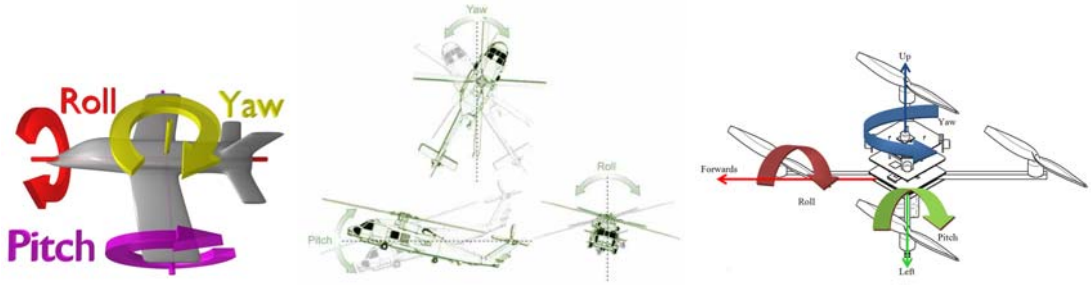


Figure 1: Pitch, roll and yaw flight dynamics

2.2. Quad Rotor Helicopters

Quad rotor helicopters are naturally very unstable aircrafts as they have no notion of aerodynamics, thus the ease of their design. However, the variety of installed sensors makes them very stable VTOL aircrafts.

Another key feature of such model is its propellers. In summary, the propellers of small quad rotor aircrafts consist of two main physical properties, their diameter and pitch. The first has to do with the required thrust for the model to hover, which again depends on the size and the weight of the UAV. The second is the vertical distance the aircraft achieves by 360^{o} rotation. For a stable model to be used in aerial photography and covert reconnaissance, propellers with large diameter and small pitch should be used. These are named "slow-fly" propellers.

A quad rotor model, as shown in the Figure 2, is cross structured [4]. It features two sets of motor-propellers. One set rotates counter clockwise (motors 2, 4) and the other set clockwise (motors 1, 3). This characteristic simply performs the role of the tail motor on the traditional helicopter models. This model structure has two main roles. It is used for balancing the aircraft, and providing the required angles during flight. These are both achieved by controlling the speed of each rotor individually and getting feedback from the installed sensors.

2.2.1 How a Quad Rotor Works

For the quad copters to achieve roll angles, the 'Right' motor (referring to Figure 2a) increases its speed while the 'Left' motor decreases or vice-versa depending on the required angle. At the same time, the other set of rotors ("Front-Rear") balances the vehicle to avoid yaw angles by increasing or decreasing speed to cancel out the torque. This is achieved due to the "left-right" rotors rotating clockwise, opposite to "front-rear" which are rotating counter clockwise. Similarly, for pitch angles the same concept applies on "Front – Rear" motors and "left-right" as balancers.

For the adjustment of the yaw angles, the whole set ("left-right") of clockwise motors is increases in speed while the set of counter clockwise motors decreases in speed to achieve clockwise angles. The opposite is applied for counter clockwise angles.

One may wonder how much each motor has to increase or decrease its speed. The answer to this is the Proportional-Integral-Derivative (PID) controller. Roughly, PID controller has an input, three gain parameters and an output in a closed-loop structure. The input of the controller is the "set point". An example can be the pitch angle a pilot sets to its quad rotor by pushing/pulling the yoke (transmitter stick) to a specified value. The PID controller is responsible to adjust the motor speeds (output) to match that angle by reading the Inertial Measurement Unit (IMU) (feedback). The gain parameters of the PID controller are three constant values which are used to define the response of the

controller in terms of rise time, overshoot, settling time and steady-state error. An in-depth analysis of how PID controllers work can be found in sub-chapter 2.3.

As shown in Figure 2b, the aircraft has six degrees of freedom (6 DOF) of according to its structure and therefore, the computational unit should be able to receive feedback of its current motion state continuously for balancing the quad and to update its attitude. For this reason, the appropriate sensors should be interfaced to generate real time commands to the rotors. The appropriate sensor for X, Y and Z vector measurements is the triaxial accelerometer. For the determination of θx, θy and θz angles, a three-axis gyroscope can be used.

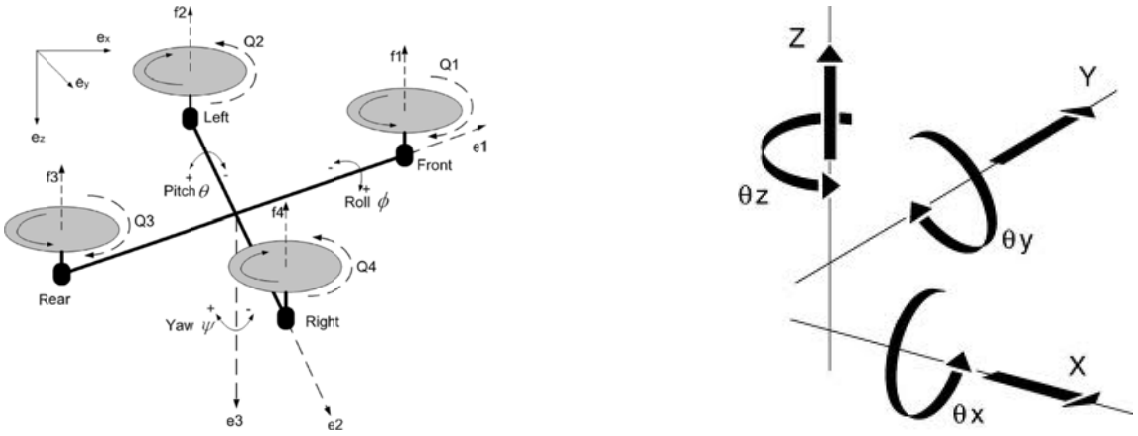


Figure 2: a) Motor structure of quad rotor helicopter **b) 6 Degrees Of Freedom**

2.3. PID Controller

PID controller is a closed loop system as shown in Figure 3a [5]. In theory, it can be applied to control any process with measurable output. The *e* variable represents the error. It is actually the difference between a 'set point R' and 'output Y'. The signal 'u' is:

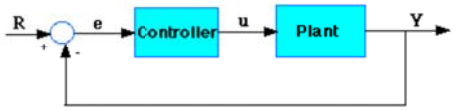
$$\boldsymbol{u = K_p e + K_I \int e dt + K_D \frac{de}{dt}} \qquad \text{Equation 1}$$

Signal 'u' is adjusting the process ('plant') and consequently a new 'output Y' is generated. It is fed back to the controller to recalculate the error *e* and then its derivative and its integral at the same time. This process runs infinitely. Each loop's iteration attempts to make 'u' equal to the 'set point R' by giving the required signal to the 'plant' to increase or decrease.

A common example for a PID controller is the central heating: The user sets the desirable temperature ('set point'). The PID controller reads a temperature sensor and calculates the error *e* (='set point' – 'current temperature'). The PID controller then

calculates the signal *u*. That signal is enough to inform the heating unit to increase power, decrease power or stop.

The same concept applies on small UAVs. The 'set point' is a reference value can be set by moving the transmitter's stick. The PID controller generates motor commands to increase or decrease their speeds. The 'current data' for error calculation is maintained by the installed sensors (IMU).



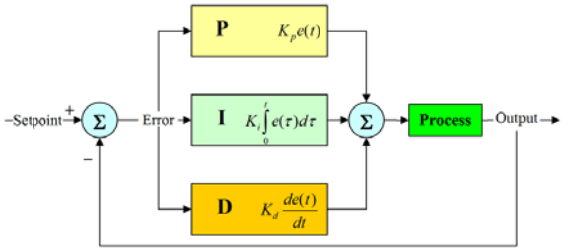


Figure 3: a) Abstract view of a PID Controller **b) The Block diagram of the Controller [6]**

2.3.1 The three parameters P, I and D

A PID controller has three gain parameters. These are constant parameters which are used to make the PID controller as fast, accurate and stable as possible.

- Kp (proportional gain parameter) is used to decrease the rise time and the steady-state error. However it cannot eliminate it. The term is given in the form of:
  P = $Kp \times e$ and has the following effect:

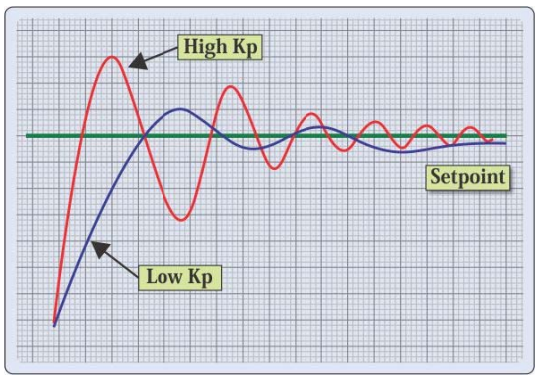


Figure 4: The "P" parameter effect

As illustrated in Figure 4, a high Kp results to a faster attainment of the system's 'set point', however it also produces an unstable system with oscillations around the set point. If the Kp is set to a low value, the system will be stable but the exact 'set point' value is not reached (stabilization below or above the set point).

- Ki (integral gain parameter) is used to eliminate the steady-state error but results in overshoot (signal oscillation around the 'set point'). The term is given in the form of:

$$\boldsymbol{I = Ki \int e}$$ Equation 2

and is effect is shown in Figure 5:

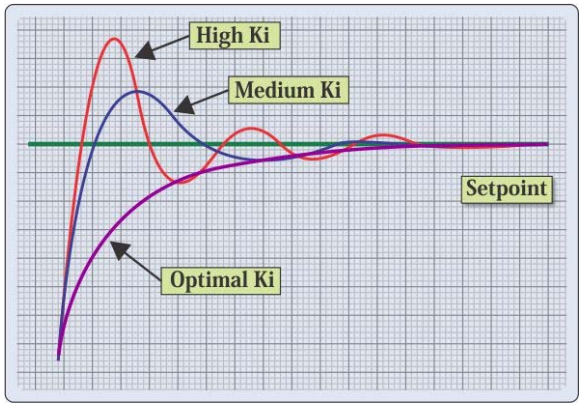


Figure 5: The "I" parameter effect

In a system with low Kp values (stable), the way Ki is affected is shown in Figure 5. The red line shows the response in the case the Ki gain selected is high. The system response is fast but also causes a large overshoot. Despite the large overshoot, the system stabilises on the 'set point' value. While the Ki gain decreases, the system response is slower but the stabilization is faster.

- Kd (derivative parameter) is used to stabilise the system. This is the last term and its given by the formula:

$$\boldsymbol{D = Kd(e_n - e_{n-1})}$$

Equation 3

This term is used to calculate the rate of change error. If the rate of change of the error is slow, the derivative gain term must be increased for the PID controller to achieve a faster response. If the rate of change of the error is fast, then the derivative gain term is decreased for the system to be stable.

A synoptic view of the effects of each gain term is shown by Table 1 follows.

Table 1: Synopsis of PID parameters effects

Parameter	Rise time	Overshoot	Settling time	Steady-state error
Kp	Decrease	Increase	Minor decrease	Decrease
Ki	Decrease	Increase	Increase	Elimination
Kd	Minor decrease	Minor decrease	Decrease	Minor decrease

The mathematical model of a PID controller's output is therefore:

$$\boldsymbol{Output = P + I + D = K_p e + K_i \times \int edt + K_d \frac{de}{dt}} \qquad \text{Equation 4}$$

2.4. UAV's sensors and orientation determination

Usually the sensors are installed on a UAV are subject to its mission activities. However, all aircrafts require some basic sensors to operate. These are: IMU, the GPS, the communication media and a processing unit. The IMU is nothing else but the combination of the sensors required to measure acceleration and the sensors required to measure changes in rotation.

For the navigation and control of the aircraft, orientation is the key role. For this to be calculated, rotation group theories must be applied on the IMU sensor measurements. Hence, the orientation of the aircraft can be represented as rotation with respect to earth [7]. That's the role of the Direction Cosine Matrix (DCM). The matrix elements can be maintained by the gyroscopes, accelerometers, the positioning system and the magnetometers.

2.4.1 Inertial Measurement Units

IMUs are responsible to provide measurements to the processing unit with respect to the aircraft's body coordinates [7]. These sensors are usually based on Micro-Electro-Mechanical Systems (MEMS) technology, hence are very small, light and low-cost. However, due to these advantages, they suffer from inaccuracy [8]. For this reason, several sensors were combined, each complementing the other, in order to provide accurate data to the control system. The most important measurements of an aircraft for stability and attitude control are the acceleration and the angular velocity in all 3 axes [4]. Acceleration in 3D space can be measured by triaxial accelerometers and angular velocity by three-axis gyroscopes. Attitude in simple models used to be calculated only by

gyroscopes. However, for a fully stabilized flight, accelerometers, GPS and/or magnetometers are required to correct the gyroscope errors.

Referring to Figure 2a , Let I = {e_1, e_2, e_3} be the earth's inertial frame, where e_1 points to magnetic north and e_3 is the vertical direction.

The centre of its body frame can be denoted as B = {e^b_1, e^b_2, e^b_3}. A rotation of R: $-\boldsymbol{sin\theta sin\varphi cos\theta cos\varphi cos\theta}$ Equation 5 represents the relation of DCM and Euler angles. Effectively, R matrix is the required data for expressing the orientation of the aircraft with respect to ground.

$$R=\begin{pmatrix} \boldsymbol{cos\theta cos\psi} & \boldsymbol{sin\varphi sin\theta cos\psi - cos\varphi sin\psi} & \boldsymbol{cos\varphi sin\theta cos\psi + sin\varphi sin\psi} \\ \boldsymbol{cos\theta sin\psi} & \boldsymbol{sin\varphi sin\theta sin\psi + cos\varphi cos\psi} & \boldsymbol{cos\varphi sin\theta sin\psi - sin\varphi cos\psi} \\ \boldsymbol{-sin\theta} & \boldsymbol{sin\varphi cos\theta} & \boldsymbol{cos\varphi cos\theta} \end{pmatrix} \quad \textbf{Equation 5}$$

The three angles are therefore roll (θ), pitch (φ), yaw (ψ).

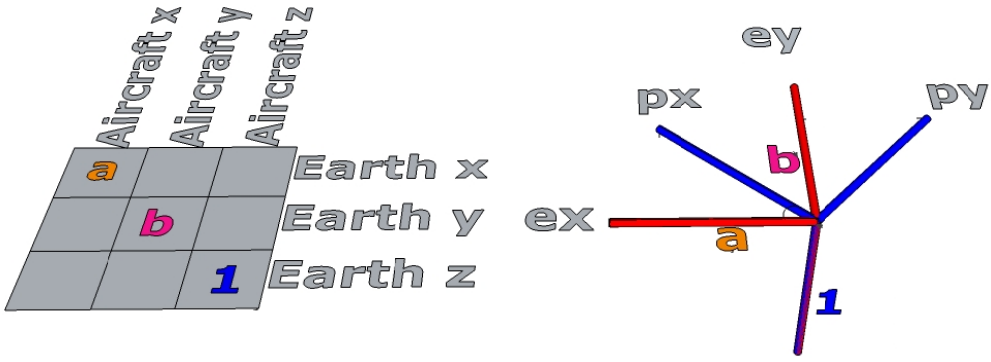


Figure 6: a) Aircraft - earth coordinate systems b) Illustrated rotation

Referring to Figure 6b, 'a' is the projection of aircraft's 'x' down on earth's 'x'. While unit vectors were used in the rotation matrix, 'a' is the cosine of the rotation angle. By calculating the dot product of aircraft's 'x' with earth's 'x', 'a' can be observed. Hence, the angle can be calculated by taking the arccosine of 'a'.

Similarly, 'b' is the projection of aircraft's 'y' on earth's 'y'. Again, 'b' is the cosine of the rotation angle as soon as unit vectors are used in the rotation matrix. As the earth's coordinate system is assumed fixed, any rotation of the aircraft's coordinate system implies that the aircraft has rotated.

2.4.2 Flight Control and DCM Relation

If the $R=\begin{pmatrix} \boldsymbol{cos\theta cos\psi} & \boldsymbol{sin\varphi sin\theta cos\psi - cos\varphi sin\psi} & \boldsymbol{cos\varphi sin\theta cos\psi + sin\varphi sin\psi} \\ \boldsymbol{cos\theta sin\psi} & \boldsymbol{sin\varphi sin\theta sin\psi + cos\varphi cos\psi} & \boldsymbol{cos\varphi sin\theta sin\psi - sin\varphi cos\psi} \\ \boldsymbol{-sin\theta} & \boldsymbol{sin\varphi cos\theta} & \boldsymbol{cos\varphi cos\theta} \end{pmatrix}$ is rewritten as $R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix}$ for simplicity, then the way the

DCM is applied on the aircraft as observed from [9] is described below:

- For the pitch control of the aircraft, the current pitch angle must be known. This can be determined by calculating the dot product of the aircraft's roll axis with the earth's 'y'. The direction cosine r_{zx} is the dot product of the roll axis with the earth's 'y'. However, r_{zx} is zero when the aircraft is in level position.
- Similarly for the roll control, the current pitch must be known. This can be found by calculating the dot product of pitch axis with the earth's 'y'. Furthermore, r_{zy} can be found with the dot product of the pitch axis with earth's 'y'. Again, r_{zy} is zero when the aircraft is in level position.
- The direction for navigation can be determined by the GPS course. Hence the yaw attitude is given by calculating the cross product of roll axis of the UAV with the desired direction vector. A test however must be made firstly to observe if the vehicle is not oriented opposite to its destination. That information is determined by calculating the dot product of the roll axis with the vector of the desired direction. That will give the course and if that is negative, the aircraft must rotate (yaw axis) more than 90 degrees to recover its correct orientation.
  As the interest is focused on the horizontal components only, the z are set to zero, therefore the vector is $[r_{xx} \quad r_{yx} \quad 0]$. The cross product of this vector with the direction wanted, gives the sine of the angle the aircraft is off course.
- The vehicle is in reverse (upside down) if the dot product of yaw axis of the vehicle with vertical axis is negative. The dot product of the aircraft's yaw axis with the earth's 'y', will give the r_{zz} of the matrix. If the aircraft is directed opposite it destination, a zero is obtained.
- The rate of turning around the earth's 'y' axis can be determined by the dot product of vertical axis with the gyroscope rotation vector. Therefore the rate of the rotation of the aircraft around the earth's y can be obtained by calculating $\omega_x r_{zx} + \omega_y r_{zy} + \omega_z r_{zz}$.

In other words, assuming the DCM is updated by the gyroscope, accelerometer and magnetometer sensors correctly, the following matrix is computed:

$$\boldsymbol{R'} = \begin{bmatrix} \boldsymbol{GyroROLL} & \boldsymbol{GyroPITCH} & \boldsymbol{GyroYAW} \\ \boldsymbol{LongitudinalAcceleration} & \boldsymbol{LateralAcceleration} & \boldsymbol{VerticalAcceleration} \\ \boldsymbol{Gravity} & \boldsymbol{MagnetometerX} & \boldsymbol{MagnetometerY} \end{bmatrix} \textbf{Equation 6}$$

Hence, the current orientation angles can be determined as:

- ROLL = atan2($MagnetometerX$, $MagnetometerY$);
- PITCH = -asin(Gravity);
- YAW = atan2($LongitudinalAcceleration$, $GyroROLL$);

2.4.2.1 Gyroscopes

Gyroscopes are required for the attitude calculation as they provide angular velocity measurements. However, the required parameters are the three angles (roll, pitch, yaw), hence an integration on the measured angular velocities must be performed [7]. A problem which arises is that MEMS IMUs are following the vehicle (they do not have moving parts as traditional gyroscopes) and produce signals proportional to the rotation rate, hence there is no change on the rotation. The solution to this problem is the calculation of the sequence of rotations. Thus, the gyroscopes can provide the desired data to the DCM.

As mentioned before, these sensors are not accurate in readings and considering the required integration, an error might grow to infinity. That is known as drift [7] .The solution to this is the recognition of these errors and the application of adjustments before the DCM is constructed. These adjustments can be made by the use of a PI (proportional-integral) controller and the use of reference vectors. This controller provides negative feedback to a drift adjustment unit ensuring the DCM would be accurate as shown in Figure 7. The most common technologies are used to connect gyroscopes to the processing unit are the Inter-Integrated Circuit (I^2C) and the analogue interfaces.

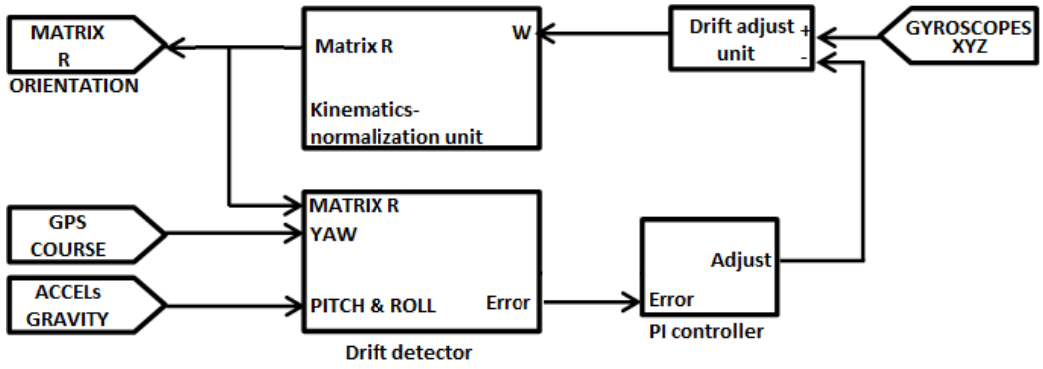


Figure 7: Orientation calculation procedure

2.4.2.2 Triaxial Accelerometers

Accelerometers provide measurements of inertial forces and gravity. Hence, the direction of gravity can be determined. Thus, it provides orientation measurements contrary to gyroscopes which provide the change to orientation. Thankfully, any error in the acceleration measurements is not as crucial as in the gyroscopes because each measurement is not affected by the previous one.

As a result, the role of the accelerometers on such vehicles is to provide the roll and pitch error corrections of the gyroscopes drift [8]. That is because the aircraft has constant forward speed and does not accelerate frequently, however, any instability in roll or pitch is measured. Accelerometers provide the reference vector Z for the gyroscope's

drift correction. Again, common interfaces for connecting MEMS accelerometers to the computational unit are the I2C and the analogue interfaces.

2.4.3 Positioning System

The most popular positioning system is the Global Positioning System (GPS). It is a satellite-based system that makes use of 24 satellites in order to provide accurate (about 3m deviation) positioning information. This information can be used as the reference vector for yaw. Hence, the yaw orientation of the model is enhanced while it cancels out the gyroscope drift. The reason the GPS cannot be the only sensor for yaw calculations is its low update rate which is normally at about 1-4Hz (update: GPS receivers with 10Hz update rate were released in the market for civilian needs at low cost after this project started)The GPS receivers provide information such as longitude, latitude, speed and course over ground and altitude. The course over ground is defined as the clockwise angle from north to direction vector.

The way to obtain information from a GPS receiver is the NMEA (0183) format [10]. This format is a human readable standard ASCII encodes delimited by commas (“,”). The interface of connecting such a device to a processing unit is usually the UART.

.

2.4.3.1 NMEA 0183

NMEA is the abbreviation word of National Marine Electronic Association [10]. It is a communication serial protocol that GPS devices are using to exchange position information. This exchange is done via well-formed sentences. The sentences are characterised by the ‘$’(dollar sign) at the start of the sentence and end with a checksum of a ‘*’ (star) and two hexadecimal numbers (ex. *4F) followed by <CR><LF> (Carriage Return and a Line Feed).

There are several types of NMEA sentences available on GPS receivers, however not all of them are providing the required information for UAVs autonomous navigation. Useful sentences for autonomous navigation are usually those they provide at first positioning information, fix quality, number of satellites connected, altitude above sea level, speed over ground and course over ground.

Unfortunately, this information cannot be extracted from a single sentence but from two. These are the GPGGA which provides all the above information apart from speed and course over ground. Such data are available in the GPRMC sentence. A sample sentence is analysed in Table 2.

Table 2: Sample NMEA sentence [11]

Sentence sample	$GPGGA, 170834, 4124.8963, N, 08151.6838, W,1,5,1.5, 280.2, M, -34.0, M,, *75	
Name	Example Data	Description
Sentence Identifier	$GPGGA	Global Positioning System Fix Data
Time	170834	17:08:34 Z
Latitude	4124.8963, N	41d 24.8963' N or 41d 24' 54" N
Longitude	08151.6838, W	81d 51.6838' W or 81d 51' 41" W
Fix Quality:	1	Data is from a GPS fix. Fix term is assures that the positioning information are correct (accurate enough for navigation).
0 = Invalid		
1 = GPS fix		
2 = DGPS fix		
Number of Satellites	5	5 Satellites are in view
Horizontal Dilution of Precision (HDOP)	1.5	Relative accuracy of horizontal position
Altitude	280.2, M	280.2 meters above mean sea level
Height of geoid above WGS84 ellipsoid	-34.0, M	-34.0 meters
Time since last DGPS update	blank	No last update
DGPS reference station id	blank	No station id
Checksum	*75	Used by program to check for transmission errors

2.4.3.1.1 NMEA Checksum Calculation/Validation

The way the NMEA sentences are checked against corruptions is the checksum value at the end of each sentence. The algorithm for validating the sentences is by performing an XOR between all bytes which exist between the '$' sign and the '*' sign.

A sample C code for calculating the checksum is:

```
int CheckSum(char *sentence) {
int i;
int res=0;
int c;
      for (i = 0; i < strlen(string); i++) {
            c = (unsigned char)string[i];
            if (c == '*') break;
            if (c != '$') res ^= c;
      }
return res;
}
```

2.4.4 Magnetometers

Magnetometers can sense the earth's magnetic field. While they are not placed near magnetic field isolators, their measures are accurate enough to provide the last bit of information required for the attitude calculation while the craft is hovering. That's the yaw angle, which can be determined by measuring the magnetic field in the body of the aircraft. In addition, after a GPS signal failure, magnetometers can still navigate the aircraft until the signal is recovered.

2.4.5 Collision/Obstacle Avoidance

In the field of autonomous UAVs, several technologies have been used for collision/obstacle avoidance. The most accurate approach is the obstacle detection using computer vision and image processing techniques [12].

Although it is an accurate solution for obstacles detection and avoidance, such a solution increases dramatically the weight and the size of the UAV as well as its cost. Consequently the flight time is reduced by a lot. This approach is often used for indoor autonomous vehicles to build a routing map or on medium to large sized UAVs. The way they work is by using stereo vision or 3D cameras to detect near objects. Both techniques are measuring distances by analysing video frames.

Another solution for obstacle avoidance is the use of range-finder sensors [13]. Two common types of these sensors which can be used are sonar range-finders and infrared range-finders. Their advantage over vision obstacle detection is their small size, weight and their low cost. Furthermore they do not require any computation on their output for observing the distance between them and the obstacle. Such devices can be connected on a microcontroller via the I^2C or analogue input interface.

2.4.6 Barometers

Barometers are sensors for sensing the atmospheric pressure. Their main role on a UAV is for altitude calculation, which is much more accurate than the GPS estimations. Altitude measurement on an aircraft is required to be transferred to the ground control station for mission planning. Furthermore, it is required for the aircraft in order to hold its altitude. In addition, a barometer provides measurements in Pascal units. Hence, the following formula should be applied for altitude calculation in meters:

$$p = 101325\ (1 - 2.25577\ 10^{-5}\ h)^{5.2558}$$

Where
p = air pressure (in Pa)
h = altitude above sea level (m)

2.4.7 Electronic Speed Controllers and Motors

Motors are commonly used on small aerial models, are usually the brushless out runner motors. Contrary to the traditional motors, the power is not applied on the shaft, but on the coils. An electronic circuit called ESC is responsible to apply the power to the coils for the generation of the magnetic field. As soon as the magnets and the shaft are permanently attached on the drum, the whole drum is rotating eliminating friction and erosion.

The term ESC stands for Electronic Speed Controller. It is a circuit that has as a high power source as input and the brushless motors as output. The amount of energy transfers from the input source to the motors is controlled by a Pulse-Width Modulation input. That structure is used for controlling high power consumption motors through a low powered circuit such as a microcontroller or a microprocessor.

2.4.8 Sensor Interfacing

Four main types of interfacing are available for connecting sensors to the processing unit. These are the PWM for the DC rotors, UART for GPS signal receivers, analogue input and I^2C protocol for the gyroscopes, accelerometers, magnetometers and ultrasonic range finders.

2.4.8.1 Pulse-Width Modulation

PWM is a digital method of controlling analogue circuits [14]. Most of the microcontrollers and DSPs feature this technology making the implementation easy. The way it works is by splitting the duty cycle of a square wave into desired-size modules. At any time of the output signal a digital value is observed, however due to the high switching speed, the overall output voltage signal forwarded to the input of the analogue-in device is seemed as analogue. Figure 8 illustrates the PWM method for a 10%, 50% and 90% output of the DC supply. For instance, if the operating voltage of a microcontroller is 5V, the output of the PWM would be 0.5V, 2.5V for the second and 4.5V for the third. The resolution of the output depends on the architecture of the embedded system is in use.

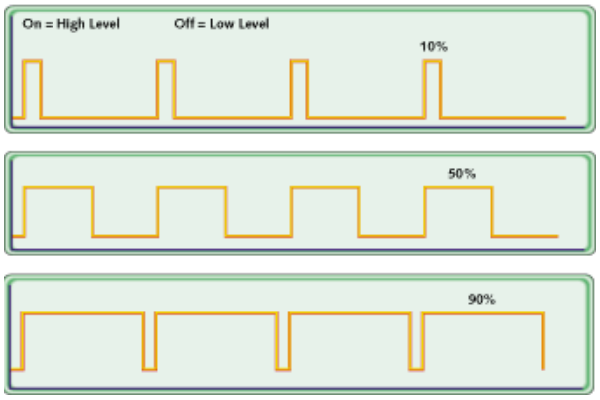


Figure 8: PWM operation

2.4.8.2 I²C (Inter-Integrated Circuit)

I²C bus offers the opportunity of interfacing up to 112 devices. According to [15], the I²C has 7 bits of address space (2^7=128) where 16 of them are reserved. It has 2 bidirectional lines, the SCL (serial clock) and SDA (serial data line). The advantage of I²C bus over the analogue input is that it delivers analogue information in a digital manner, hence interfacing devices that accuracy matters. I²C ensures a noise-free transmission to the destination. Every device connected on the bus has a unique address and can operate either as a receiver or a transmitter. That address has to be used in order to a device can communicate with another device. Figure 9 illustrates the required connection between the devices.

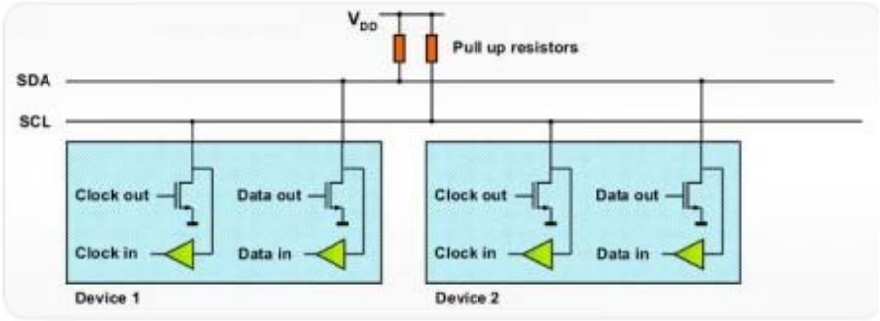


Figure 9: I2C bus with 2 devices

Communication procedure:

- The master ensures the bus is free (Both lines are high)
- The master sets the SDA line to low and then sends the receiver address

- The Read/Write request bit is sent after the 7 bit address
- Slave recognizes its address and acknowledges by setting SDA low at the 9th SCL cycle.
- The data is sent
- Acknowledgement is sent
- When SCL is high and SDA has a rising edge the stop condition is indicated.

2.5. Wireless Communication

Most common types of communication media found on small UAVs are the wireless 802.11x and RF technologies [16]. These types of communication provide a high rate of data transfer as well as reliability in terms of successful transmit/receive of data. As the payload of the UAVs is limited, the communication equipment they can carry has to be limited in weight; as a result the effective range of communication between the ground control stations is restricted to several miles.

The communication between the UAV and the ground control station has to be bidirectional. The ground station is sending navigation orders while the vehicle is sending flight data back to the ground station. For this reason wireless communication media are used such as Wi-Fi which features the reliable TCP/IP protocol [16].

Another wireless communication media is the GPRS/3G which is a GSM service [17]. It uses the GSM cellular network to exchange reliable data using TCP/IP transport layer protocol. All well-known application layer protocols such as HTTP, POP3, FTP etc. are working on top of it.

3. THEORETICAL BACKGROUND ON GCS

As UAVs are built to execute missions, a ground control station is required for real time control commands on flight directions and on camera rotations. As the UAVs offering autonomous flights, the GCS should provide all the information can be found in manned air vehicles as well as an interface for:

- Configuration of the UAV's missions
- Real time tracking of the vehicle over the globe
- Monitoring instrumentation and status of flight
- FPV (first person view) live video streaming
- Mission alterations and corrections
- Aircraft initialisation procedures

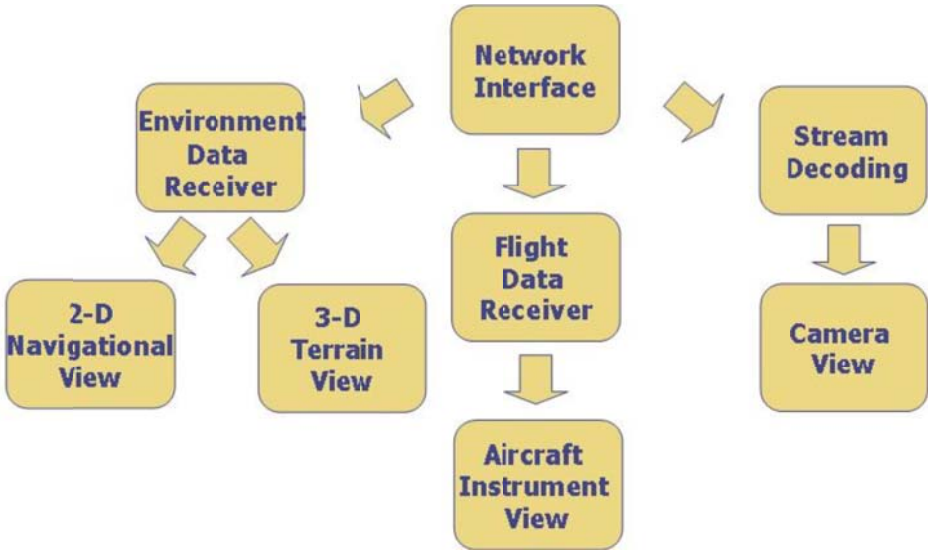


Figure 10: structured components approach of a GCS software [18].

The software system continuously receives raw data from the aircraft. This data is processed by different units on the system to generate a visual representation of the aircraft's state. The raw positioning data are adapted in a way to provide real time route tracking and orientation. Furthermore, the measurements of the aircraft's sensors are converted to an artificial horizon where the operator has a notion of the flying state of the UAV. Apart from that, the graphical interface provides monitoring of the aircraft's vital components for any failure avoidance via the artificial instrumentation.

3.1 Tasks and Instrumentation of a Ground Control Station

The user should be able to plan a mission by specifying the coordinates of each destination as well as the priority of each. The system is also responsible to perform the pre-flight checks before proceeding to the take-off action [18]. During the flight the user is monitoring the aircraft's orientation as well as the mission progress. That is achieved soon as the GCS is continuously transforms the data are coming from the UAV to artificial instruments. This instrumentation helps a user to have notion of the flight status. The main instruments should exist on a ground control station are:

- Real time GPS Information
- Altimeter
- Artificial horizon / Head-Up Display (HUD) is shown in Figure 11
- Battery/Fuel status
- Speed over ground
- Air speed

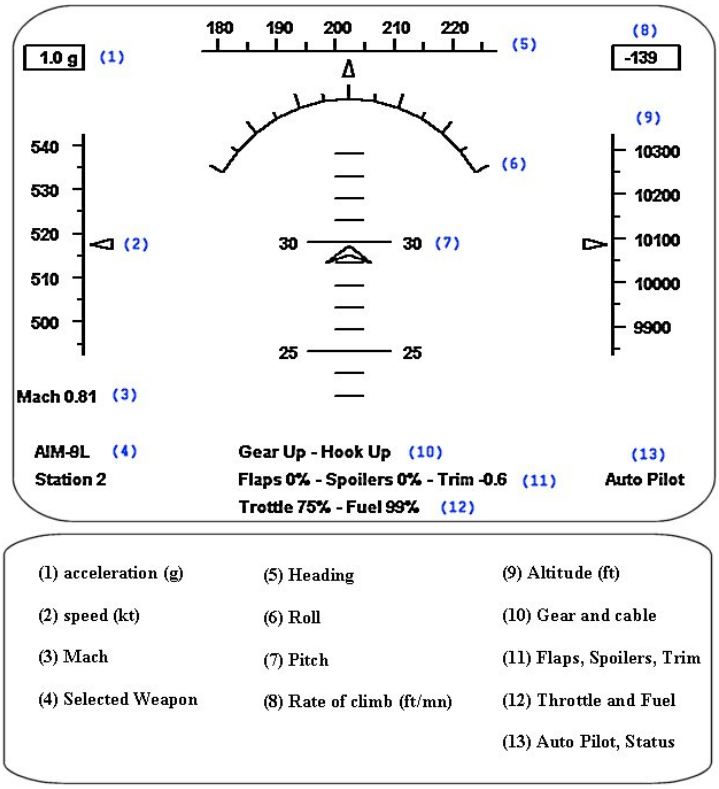


Figure 11: A sample HUD for visualization of flight data [19]

The GPS information are available within a GCS is used for the determination of the current position of the aircraft with respect to its destination. Hence, the UAV can be tracked on a map. The HUD displayed in Figure 11 is used for the determination of aircraft's orientation. This information combined with the current positioning information

help the user to inspect the trajectory of the UAV. Furthermore, information such as the altitude, speed and fuel are shown

3.2 Mission planning and navigation overview

The GCS is responsible for the mission monitoring and planning. It should provide the user with a virtual globe to plan the desired missions. After the user selects the destinations of interest, the software should calculate the routes and estimate the flight time. Through the flight the user can amend any mission or direction deviation as the current position of the aircraft is known.

3.3 Communication

A communication protocol must be agreed between the UAV and the ground control station. The information to be sent consists of two types, the main and secondary flight information. A dedicated mechanism receives and handles that information in real time. The data acquisition unit is responsible to process the information and transfer it modularized to the respective units to perform the visual presentation. A sample well-formed message is shown in Figure 12 [18].

Basic message format

SOM	LEN	TYPE	DATA	CSUM	EOM

TYPE 0x02 - basic flight data
DATA field contents:

Byte	Description
1:4	GPS latitude
5:8	GPS longitude
9:14	GPS height
15:16	GPS ground speed
17:18	GPS azimuth
19:20	Air speed
21:22	Barometer height
23:24	Motor revolution
25:26	Radar height
27:28	Compass reserved
29:34	Horizon reserved

TYPE 0x04 - secondary flight data
DATA field contents:

Byte	Description
1:2	Fuel level
3:4	Fuel consumption
5:6	Engine temperature
7:8	Engine exhaust gases temperature
9:10	Ambient temperature
11:12	Temperatue inside aircraft
13:14	Batery level
15	Number of visible GPS satellites
16	Number of GPS satellites being tracked

Figure 12: Sample packet format

The information sent from the aircraft has to be encoded in a packet. This packet should have an overhead containing a 'start-of-message' field, 'the packet length' and 'type'. Then the significant information follows depending on type:

- Main flight information:

    a. GPS position
    b. Speed over ground
    c. Course over ground
    d. Altitude
    e. Compass measurement
    f. Horizon information

- Secondary flight information

    g. Battery level
    h. Number of GPS satellites connected to
    i. GPS signal lock indication (GPS fix)

Apart from that, the GCS provides video streaming for surveillance. The streaming information might be sent either through the communication link or by a dedicated remote long-distance surveillance unit. In both cases, the GCS is responsible to combine the video information with artificial instrumentation discussed in chapter 3.1 Tasks and Instrumentation of a Ground Control Station.

4. IMPLEMENTATION

As stated above, the work made is divided into two parts. The work made on the UAV and the creation of the GCS software. The sub chapter 4.1 is dedicated to the design and implementation of the UAV while sub chapter 4.2 to the development of the GCS.

4.1. UAV Design - Hardware

The UAV design is further divided into two main parts: the hardware design of the aircraft and the software integration of the current code to meet the objectives of the project. For the hardware part, a PCB designed to interface all the electrical and electronic components required. Therefore, the existing software of the remote controlled quad copter integrated to meet the objectives of the project listed in sub chapter 1.1.1.

4.1.1 Frame design

The frame of the UAV is designed by following the guidelines of the open source quad rotor helicopter [1]. Its setup selected to be medium sized as it offers stability over wind, further payload capabilities and long flight times. Its final weight is 1.8 kg and its flight time is between 17-25 minutes. The time variation depends on the flying altitude and the wind as both variables affect the power consumption of the motors.

4.1.2 Hardware/PCB design

The PCB design and implementation was a result of studying the sensors and the computational unit of the quad rotor's architecture mentioned before. For the introduction of new electronic components to enable the internet connectivity and autonomous navigation, the following microcontroller specifications were considered [20]:

- 8-bit AVR RISC-based
- Flash memory: 256Kb
- Clock: 16MHz
- I/O pins: 86
- I^2C protocol
- UARTs: 4
- ADC channels: 16 (10 bits resolution)
- SRAM: 8Kb
- EEPROM: 4Kb
- PWM pins: 15 (8bit resolution)

Figure 13 shows the block diagram of the PCB design. The sensors, GPS/GPRS module and RC receiver are the inputs to the system while the ESCs and some indicating LEDs are its output. The ESCs are connected with the microcontroller through the PWM protocol. These are responsible to drive the high power motors as described in the chapter 2.4.7.

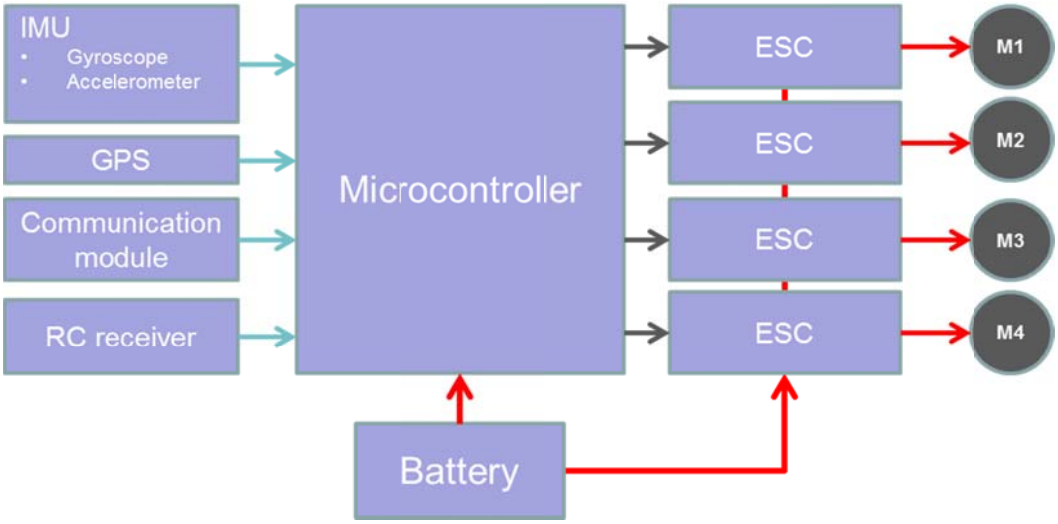


Figure 13: Control Block diagram

Based on these specifications, a new PCB was designed and implemented featuring all the sensors were already supported by the software but what is more, a new GPS/GPRS module. This adds to the system the required resources for the UAV-GCS communication as well as positioning information for autonomous navigation. Figure 14 shows the evaluation board used as computational unit. As shown, it includes the microcontroller chip as well as circuits for wired communication. Most of the specifications mentioned above are also visible on the figure.

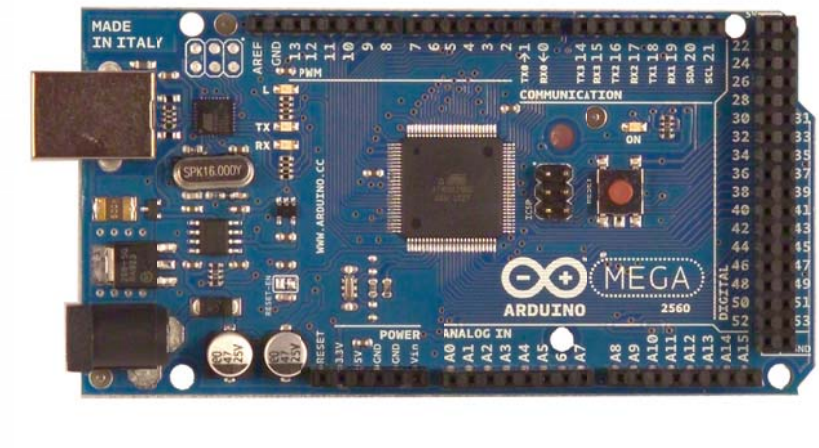


Figure 14: Microcontroller board used as computational unit [21].

4.1.2.1 Sensors Overview

Figure 15 shows all the sensors used for the implementation of the IMU and the communication link. By using these sensors, the first task to keep the price low is achieved. All the sensors apart from the GPS/GPRS module were interfaced with the microcontroller using the I^2C protocol. For the GPS/GPRS module interfacing, UART communication protocol was used.

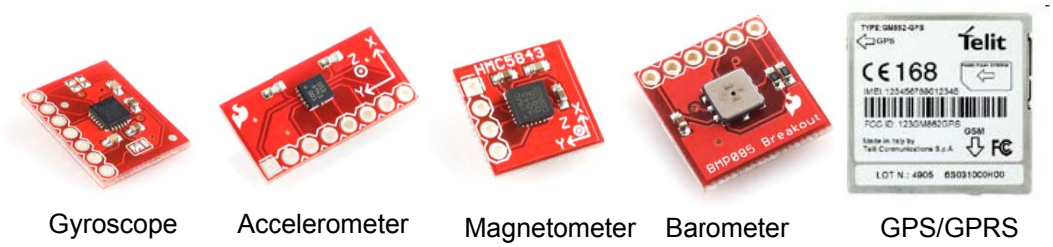


Figure 15: Sensors used

- Gyroscope specifications:
  - 3 axis measures (X, Y and Z)
  - 16 bit resolution
  - Low Pass filter
  - Low current consumption (6.5mA)
  - Power supply range: 2.1V – 3.6V
  - I^2C interface (400KHz)

- Accelerometer specifications:
  - 3 axis measures (X, Y and Z)
  - Measure ranges: ±1g, 1.5g, 2g, 3g, 4g, 8g, 16g
  - 14-bit resolution
  - I^2C interface
  - Power supply range: 1.2V - 3.6V
  - Low current consumption (0.65mA)
- Magnetometer specifications:
  - 3 axis measures (X, Y and Z)
  - I^2C interface
  - 12-bit resolution
  - Power supply range: 2.5V - 3.3V

- Barometer specifications:
  - Measuring range of 300 to 1100 hPa
  - I^2C interface
  - Power supply range: 1.8V - 3.6V

- Pressure resolution readings: 19-bit
- Temperature resolution readings: 16-bit

- GPS/GPRS module specifications:
  - SiRF III High Sensitivity GPS Receiver
  - Embedded TCP/IP Stack
  - GPRS Class 10
  - AT commands for control
  - UART line for AT commands
  - UART line for NMEA data (dedicated GPS UART)

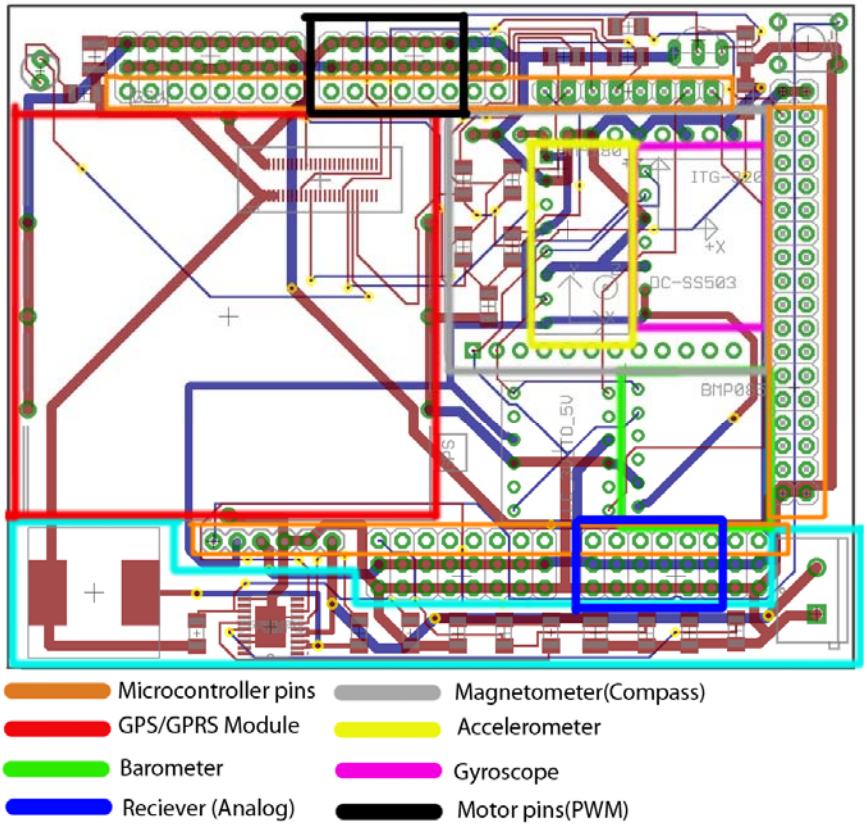


Figure 16: The new PCB designed

The layout of the new PCB (Appendix chapter 9.2) is shown in Figure 16. It fits on top (orange rectangles) of the microcontroller shown in Figure 14 interconnecting all the sensors shown by different colour rectangles with the microcontroller. The cyan shape shows the power supply designed for powering the GPS/GPRS module which works on different voltage levels compared to the sensors and the microcontroller.

All the remaining sensors (barometer, accelerometer, gyroscope, magnetometer), are connected to the microcontroller using the I^2C interface. The way the protocol works

is described in the theoretical background 2.4.8.2 [15]. The GPS/GPRS (red rectangle) is connected with microcontroller using UART interface after the appropriate voltage level conversions applied. The schematic of the PCB design detailing all the work done with hardware PCB design is included as appendix.

The RC receiver is connected through the analogue pins shown in deep blue rectangle while the ESCs are connected on the PWM pins shown in black rectangle to drive the motors.

4.1.3 The UAV Assembled

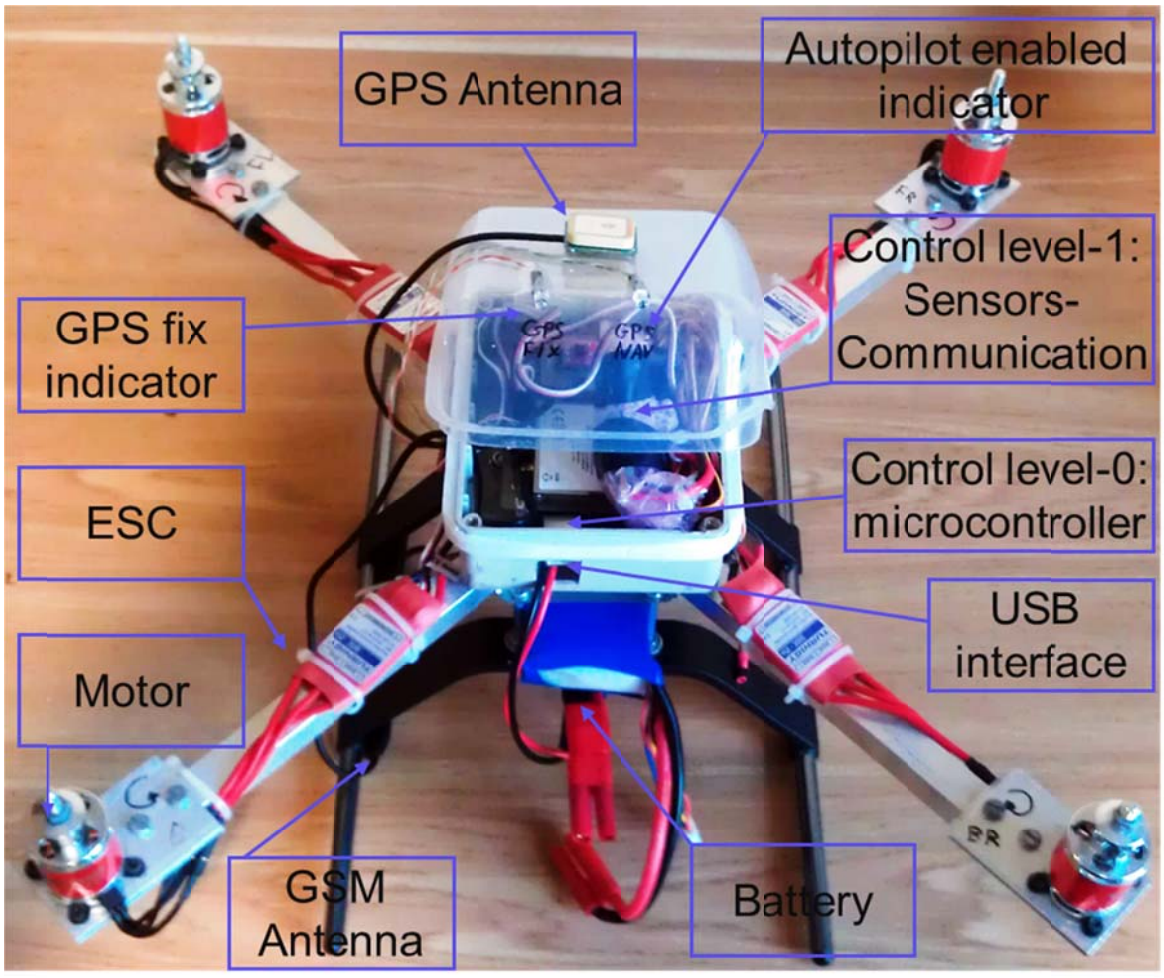


Figure 17: The UAV assembled

Figure 17 is an image of the UAV model implemented. It shows how the hardware components are interconnected. The battery installed is a 5Ah Lithium Polymer battery able to provide up to 100A current to the motors. It voltage is 11.1V. It also powers up

the control circuit. The ESCs inputs are the battery and the PWM outputs of the microcontroller. Their outputs are connected to the motors as shown in the Figure 13 block diagram. The Control circuit is divided into two levels. The microcontroller evaluation board is installed at the bottom while the PCB with the required sensors is attached on top of it. The GPS antenna is installed on the top of the cover as it requires light-of-sight view with the sky for better signal reception. The GSM antenna is mounted as far as possible form the electronic components because various tests showed that interferes with a sensor. Last but not least, the USB port mentioned provides access to the microcontroller for programming, debugging but what is more, wired connectivity with the GCS for pre-flight configuration and wireless communication initialisation.

4.2 The UAV Software

The software of the UAV developed performs two tasks. At first, it enables the vehicle to navigate itself (autopilot) while the same time enables wireless connectivity with the GCS platform through internet.

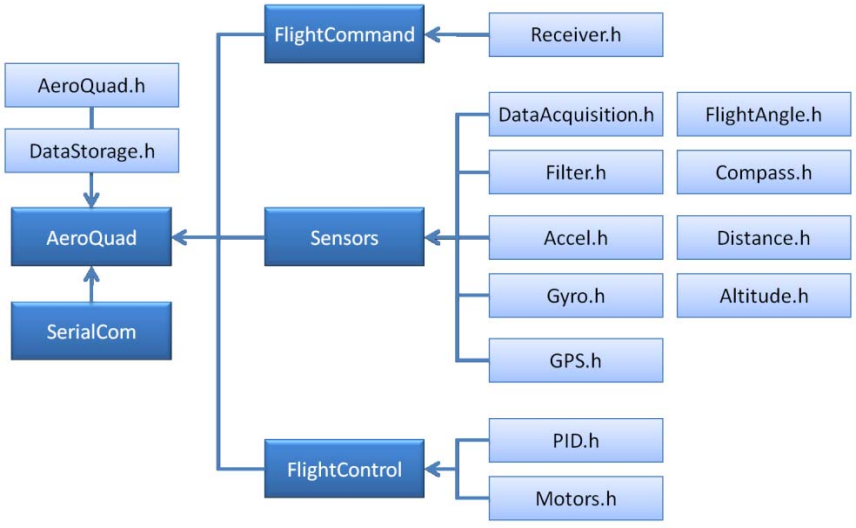


Figure 18: The Block diagram of the open source software already exists [1]

Figure 18 represents the structure of the software the quad rotor vehicle use for the RC navigation [1]. After the study of the software, several observations were made about the navigation control lead to the solution approach:

- There is a DCM filled with data of all sensors as described in "Theoretical background on UAVs" chapter 2.4.2.
- There is a PID controller for each angle of the aircraft dynamics (pitch, roll and yaw).

- A PID controller update function exists which takes three parameters to update the aircraft's orientation: the "set point", the "current data" and a pointer to the object describing each angle (this object contains information such as P, I, and D gain parameters described in chapter 2.3.1).
- The "set point" of each PID controller is the transmitter's input.
- The "current data" of each PID controller can be maintained by the DCM:
    - For the ROLL = atan2($MagnetometerX$, $MagnetometerY$);
    - For the PITCH = -asin(Gravity);
    - For the YAW = atan2($LongitudinalAcceleration$, $GyroROLL$);
- There is a function takes the PID controllers' output and generates motor commands.
- The control loop runs at 100Hz (10ms update rate).

The following statement is an abstract approach of what made for the autonomous navigation of the aircraft:

The ideal algorithm for autonomous navigation should use the GPS data and the magnetometer readings in a way imitating the transmitter inputs which are currently assumed by the software as "set points" for the three PID controllers already exist.

Hence, an abstract block diagram of the modified flight software is presented in Figure 19. The data are sent to the internet, as shown will be delivered to the GCS for processing. The procedure is discussed in chapter 4.3.

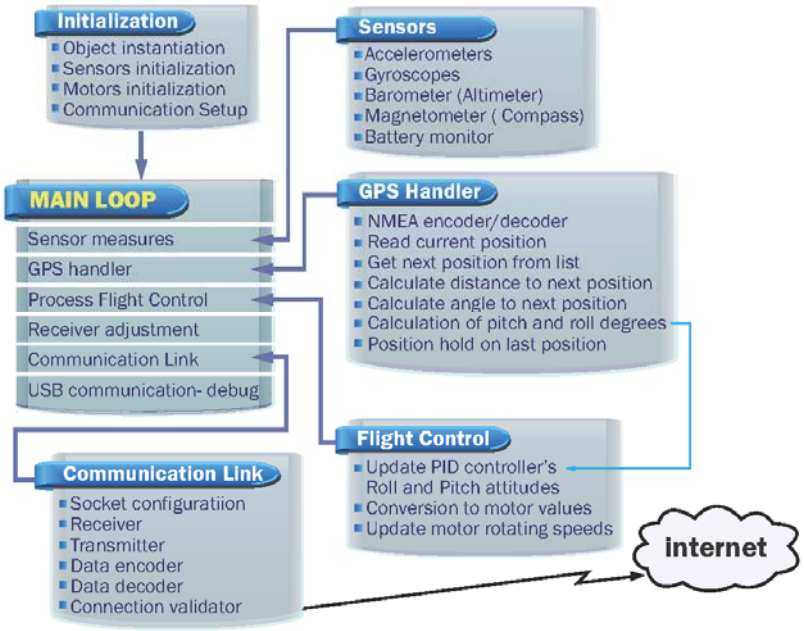


Figure 19: The block diagram of the flight software after integration

4.2.1 Autopilot Algorithm Approach

As soon as the GPS has measurable output (position coordinates), two new PID controllers were applied in a way their output being the "set point" of the three PID controllers already used by the current software (transmitter navigation) [22]. The solution approach is illustrated in Figure 20.

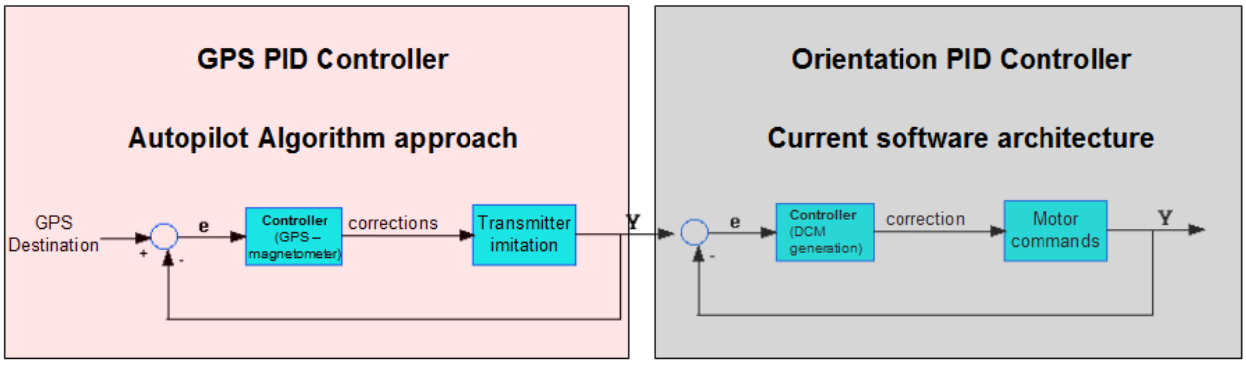


Figure 20: Block diagram of solution approach

As stated in the chapter 2.4.3, the GPS lacks of low update rate. Thus, the GPS is impossible to navigate the UAV by itself, especially when taking into account the inaccuracy of the positioning data it provides which is about 3-4 meters of deviation or worse.

For this reason the autopilot algorithm uses the magnetometer (compass) for the autonomous navigation. In summary, the way it works is by calculating the angle from the current position to the destination using the current GPS data. As the destination is set by the user and the current position is refreshed every second, the angle to destination can be observed. Another parameter is the azimuth, which can be also determined using the magnetometer. Once the required angles are calculated, the native PID controllers are updated to convert the required angles to motor commands.

The first integration made on the current open source software is the introduction of two new PID controllers, one for the pitch axis and the other for the roll axis angles dedicated to the autonomous navigation. The output of these two controllers is the 'set point' of the PID controllers used to navigate the vehicle by the transmitter. For safety reasons, the two outputs are summed up with the transmitter's command. As soon as the transmitter is not used (idle), the aircraft will be navigated by the GPS and the magnetometer. Furthermore, this approach gives immediate manual control to the UAV's orientation as soon as the transmitter commands overcome the commands generated by the autonomous navigation algorithm. To ensure that the transmitter can always overcome the algorithm's commands for safety reasons, the algorithm's value

generations are constrained to navigate the vehicle very slowly (about 11 km/h) while the transmitter can navigate it at about 30-35 km/h.

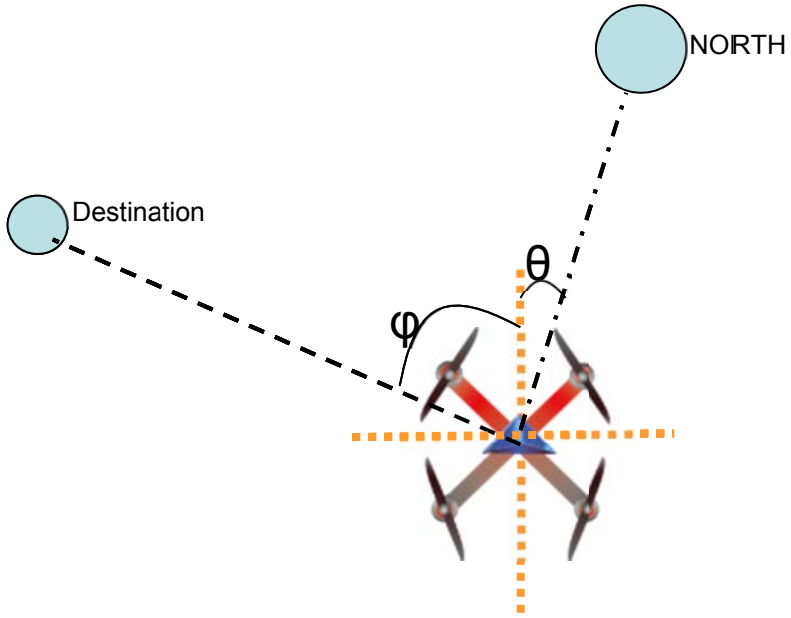


Figure 21: Illustration of the PID's "set point"

Consequently, the flight control algorithm has two roles. It has to provide the required parameters for the PID controllers update ('set point' and 'current data') as well as to check if the destination is reached. For the calculation of the 'set point', two parameters are required. The first parameter observed is the azimuth, which is the θ angle in Figure 21. That is basically a numeric value in degrees of the deviation of the aircraft's front projection to the Earth's North. This can be easily maintained by requesting a reading from the magnetometer (which the magnetometer chip's 'front' is matched the aircraft's "front"). The second parameter required is the GPS course, which is the φ angle in Figure 21. That is the clockwise angle between the direction vector of the aircraft and Earth's north. That vector is defined as $\overrightarrow{AB}$ where A is the current position coordinates and B is destination coordinates.

Trigonometry applies for the determination of this angle as follows:

- Subtraction of the last Longitude data the GPS has provided from destination's Longitude (DestinationLongitude - CurrentLongitude). The result should be then multiplied by the constant 0.649876. The multiplication converts the result in centimeters as the earth is an ellipsoid object (WGS84 ellipsoid parameters) enabling the trigonometry functions to be applied.
- Subtraction of the last Latitude data the GPS has provided from destination's Latitude (DestinationLatitude - CurrentLatitude). The result should be then multiplied by the constant 1.113195. The multiplication converts the result in

centimeters as the earth is an ellipsoid object (WGS84 ellipsoid parameters) enabling the trigonometry functions to be applied.

- The inverse tangent of the two differences results the angle to destination (φ angle): φ=`atan2(LongitudeDiff,LatitudeDiff)`.

The subtraction φ-θ gives the "target angle" which will be used as the "set point" for the PID controller updates (both pitch and roll) such as:

The sine of the target angle is the "set point" for the roll's PID controller while the cosine of the target angle is the "set point" of the pitch's PID controller.

During the calculation of the "set point", another variable calculated is the distance to destination. This is easily maintained by the use of the differences already calculated. The Pythagorean Theorem is applied for the distance calculation. $c^2 = a^2 + b^2$ where '*a*' is the `LongitudeDiff` and '*b*' is the `LatitudeDiff.`

As the "set points" are determined, the "current data" is the second parameter the PID controller requires. For the determination of this, two consecutive samples from the GPS data are required. This is the previous coordinate position (longitude, latitude) and the current position. A similar procedure is applied for the calculation as before:

- Subtraction of the current Longitude read by GPS from the very previous Longitude read depending on the GPS rate (PreviousLongitude - CurrentLongitude). Note: for the "current data" calculation the Previous Longitude is used instead of the destination's Longitude.
- Subtraction of the current Latitude read of GPS from the very previous Latitude read depending on the GPS rate (PreviousLatitude - CurrentLatitude). Note: for the "current data" calculation the Previous Latitude is used instead of the destination's Latitude.
- The inverse tangent of the two differences is the trajectory deviation angle (φ angle) which is effectively the error parameter of the PID controller: φ=`atan2(LongitudeDiff,LatitudeDiff)`

The subtraction φ-θ gives the "target angle" will be used as the "current data" for the PID controller updates (both pitch and roll).

What effectively is achieved here is the correction of the GPS deviation affecting the trajectory of the aircraft to the destination. This attempts to correct these errors because the calculation of the "target angle" is a result of both GPS and magnetometer readings (φ-θ, where θ is the azimuth obtained by the magnetometer). As the magnetometer's update rate is really high compared to the GPS's (100Hz versus 1Hz) the GPS errors are "smoothed" by the PID controller because the magnetometer provides fast and accurate corrections.

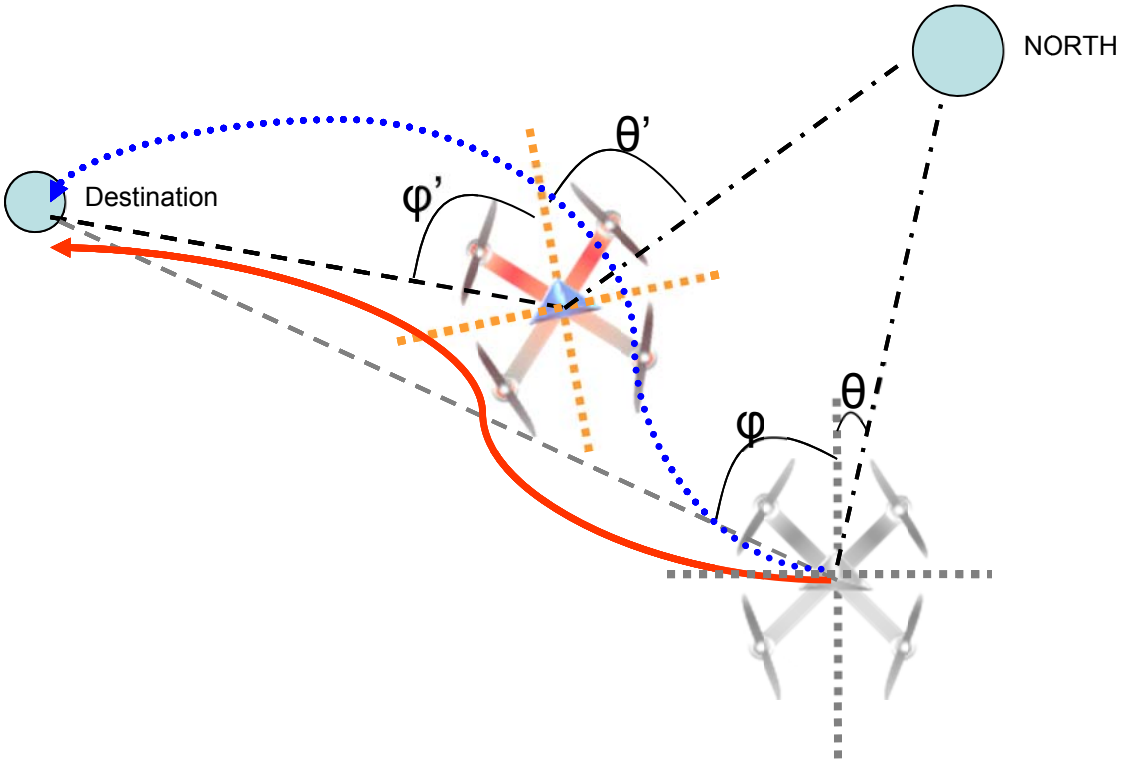


Figure 22: Autonomous navigation PID controller effect

That procedure is visualized in Figure 22 where the blue dotted line represents the trajectory the aircraft would follow if there were not any corrections. The red line represents the correction by the PID controller using the magnetometer readings. The red line is the actual trajectory the aircraft follows.

The final check which the algorithm performs is if the destination has been reached. The destination is assumed as reached if the current position of the GPS is within a radius of about 20 meters away from the actual destination position set by the user. For the calculation of this, the parameter is checked is the distance to destination compared to a value of the square root of 300(about 17.3 meters). If the destination is reached then the algorithm checks if there is another position in the mission list (positions). If there is, then the same procedure applies until all the positions are reached.

Another objective achieved which is related to the navigation algorithm, is the position hold. Once there are no more positions in the list, the roll and pitch "set points" are set to zero. Due to the inaccuracy of the GPS, the aircraft might move out the 20 meters of destination deviation. The distance will then become greater than the square root of 300 and then the algorithm is enabled again to bring it back to the destination.

Last but not least, one more objective achieved is that the autonomous fight software is independent of the communication link. Once the user has correctly uploaded the position list (missions) before flight, the algorithm ensures that the aircraft will

traverse all the positions. It will also perform a position hold (hovering) on last position. The last position should be the base.

4.2.2 Serial communication

The open source software of the quad copter features serial communication via the USB port. A command set exists which the microcontroller can understand and provide responses. Such responses are current PWM motor values, sensor readings, RC receiver readings et cetera. These commands are used for the configuration of the vehicle for RC transmitter flights. It also provides responses for debugging and observation of the current motion state of the model. Through the serial communication, the user is can configure the P, I and D parameters of the PID controllers as well as calibrate the RC transmitter and the sensors.

This command set has been extended. Further commands were added for the internet link initialization and the debugging of the autonomous navigation algorithm. For the internet link initialization, the command set extended to allow the system to read the domain of the GCS and the port listens on. That information is required for the communication handler. The extension made for the autonomous navigation debugging, is the introduction of new commands for presenting the GPS receiver's data but what's more, to report PID controllers output and their effect on motors, as described in chapter 4.2.1.

4.2.3 Communication Link Establishment

Another integration made on the current software was the introduction of internet connectivity. For the achievement of this objective, three main parts were developed. At first, an interface was developed enabling the microcontroller to communicate with the GPS/GPRS module. This also provides the required initialisation and status checks. Then, a communication object settled, providing functions for internet link establishment and information exchange. Last but not least, a new frame loop running at lower frequency than the main loop runs was developed. Its responsibility is to send the current flight data and receive any available instructions coming from GCS. That runs without affecting the stability of the flight and makes use of the communication object.

However the explanation follows of each part, is in reverse order for better understanding. It starts from the most abstract part which is the low frame loop while ends to the interface developed for the microcontroller-GPS/GPRS communication which is the lowest level.

4.2.3.1 Embedding communication feature in the main loop

A one-hertz loop frame introduced to handle data transmission and reception at a lower rate than the control loop. There was two reasons for which the communication could not be done on higher frequency. The first reason was to avoid overloads on the buffer of the GPS/GPRS module. The other reason was the GPS update rate. As soon as new data were available every second, there would be duplicates by transmitting at a faster rate. Furthermore, the less processing spent on data transmission, the faster the main control loop runs at.

The introduction of the frame loop was quite easy procedure as the current software was already using a time variable for time notation which is also required in various calculations such as the PID controllers.

The loop frame developed was placed in a position ensuring that the sensor readings are recent by the time is executed. It then performs two tasks. At first it encodes all the sensor readings in a message and requests a transmission from the communication handler. The encoding process produces a message similar to the one found in chapter 3.3.

That message has the following fields in order:

- Start of message indicator
- Current roll angle
- Current pitch angle
- Current GPS Latitude
- Current GPS Longitude
- Number of satellites in view
- GPS fix
- GPS speed over ground
- Battery monitor (fuel)
- Magnetic heading (compass)
- Altitude above/below take-off position in meters
- End of message indicator

The second task it performs is a check whether there are any pending data on the receiver buffer. In that case, it requests decoding from the communication handler.

4.2.3.2 Communication Handler

The communication handler developed is the most fundamental component for the communication between the UAV and the GCS. It makes use of a GPS/GPRS object to perform the link establishment and the exchange of information.

The responsibilities of the communication handler are:

- Initialisation of GPS/GSM module.
- Communication link initialisation.
- Reception/Transmission of data.
- Data decoding/parsing.
- Mission updating.

For the initialization of the GPS/GPRS module, the communication handler instantiates the communication object. The communication handler cooperates with the serial communication by using its capabilities of reading the GCS domain information as well as the port is listening on. By using that information, it provides to the communication object the required information to initialize the internet communication link.

Another main task of the communication handler is the management of the bidirectional Internet data link. It accepts requests for data transmission from the communication loop discussed in chapter 0 and executes them. In the case of data reception is requested from the main loop, the handler's responsibilities are more complex. At first performs the appropriate checks to decide whether the incoming data are a "mission plan" or a "mission amendment". If a mission plan message is received, it appends it to the mission list where the autonomous navigation algorithm reads the positions to fly to one by one. If a "mission amendment" message is received, the communication handler immediately replaces the position the UAV is travelling to, to the position just received from the GCS. The handler finally acknowledges the GCS that has successfully received the mission message sent.

4.2.3.3 The Communication Object

The GPS/GPRS module provides the media of Internet connectivity through the GPRS service and the TCP/IP stack; as a result the first step was to make these resources available to the microcontroller. The GPS/GPRS module is controllable via AT commands [23]. These commands are sent to the UART line and the module receives them, decodes them and finally executes them. Hence an object (class) describing the GPS/GPRS module was created. Then, some methods were constructed to translate the microcontroller requests to AT commands as well as transfer them to the UART line. These methods are also responsible for handling the responses of the module and reporting them back to the microcontroller.
The interface developed provides the following functionalities to the microcontroller:

- Toggling on/off the GPS/GPRS module
- Initialization of the module:
    - Configuration of the UART communication baud rate.
    - Configuration of the NMEA UART communication baud rate.
    - Sets PIN code of the SIM card.
    - Enabling/disabling NMEA sentences for autonomous navigation.

- Perform GSM connection.
- Perform GPRS initialization.
- Opening Internet connection (socket dial) to a specified host.

Almost every class in programming contains a constructor. The communication's object constructor is responsible to define the baud rates of both UART ports as well as the selection of the NMEA sentences will flow on the GPS UART. It also performs the GSM connection, which is required before the GPRS service can be used. Finally, the communication object is responsible to open the network socket (connection to the GCS) as soon as the domain name and port is given.

Using this approach, what is achieved is the independence of connectivity errors with the flight loop (which is the most vital component). That is achieved because the microcontroller instructs the module to perform a task. The control loop does not wait for a response, but the response waits to be read. If there is a failure within the module and the microcontroller receives no response after certain time has elapsed, the module is reset.

4.3 GCS Solution approach

Since the UAV assembled, programmed to fly autonomously and able to establish internet connectivity, an interface required for the user to configure it and plan the missions. Hence, a GCS was developed which provides to a user an interface to organize the flights. The software's structure is similar to the modularized model appears in chapter 3. The way it works is by encoding the visual information provided by user into messages and then transmits them using the communication link. On the other hand, it decodes the data coming back from the UAV and allocates them to the responsible modules for visual representation.

The software has three main modes. These are the configuration, the mission planner and the flight data. The configuration mode provides the tools for the wired connection and vehicle initialisations. The mission planner mode is an interface for planning the flights of the UAV and the flight data mode is the visual representation of the vehicle's instruments as well as it track on the globe.

4.3.1 Communication link handler

There are essentially two communication handlers in the GCS software, one for the wired and one for the wireless connectivity. The first it uses serial protocol to communicate with the UAV while the latter uses the Internet.

4.3.1.1 Serial communication handler

The serial communication handler is responsible for the managing the data coming from the serial port (USB) form the vehicle and the data are sent out to the vehicle. As it has been developed in an object-oriented programming language, it makes use of the available libraries to decode the messages and transform them to visual objects. Such objects are the HUD and motor speed indicators. The configuration tools for sensor calibration and internet communication initializations are also sent to the vehicle via the serial communication handler, just after they are formed according to the command set the serial communication 4.2.2 of the vehicle understands.

4.3.1.2 The Internet Communication handler

While the serial communication handler is manipulating the wired communication, the internet communication handler has a hardest role. Contrary to the serial communication handler, the data coming from the UAV via the internet are asynchronous. Furthermore, on the serial port there is a hardware buffer, something does not exist for the internet communication.

For this reason, an asynchronous socket listener server was developed. Once a client (UAV) gets connected, an asynchronous call back worker (similar to software interrupt) is created. The way it handles the asynchronous data is by creating a new thread each time a new byte is received. The new byte is decoded using the UTF-8 character set, and then appended to a byte list. Every time a byte is added, a check is made which decides whether the byte list contains a complete message. If a message is recognized, then the useful information are extracted and allocated to the corresponding views for visual representation. The byte list is then reset to handle the next message.

4.3.2 The Configuration View

The configuration view of the GCS provides the tools for the aircraft initialization. It has two sub views: the visual view and a command line console. It uses the serial communication handler to exchange information with the UAV via the command set described in chapter 4.2.2.
Within the configuration view, the following can be achieved:

- Communication with the UAV using the USB cable
- Initialisation of the GPS/GPRS module
    - The GCS at first instructs the UAV to enable the module
    - It supplies the domain and the port is listening information (automatically detected by software if there is internet connectivity by the time the application runs)

- Instructs the connection to the GSM service.
- Instructs the Vehicle to connect via Internet

- Observation of the mission plan is currently in the UAV's memory
- Calibration at the level position (sensor calibration)
- Clear the mission from memory
- Read the data coming from GPS receiver
- Read the pitch and roll values generated by the autonomous navigation algorithm
- Read the P, I and D parameters of the autonomous navigation PID controllers
- Set new P, I and D parameters for the autonomous navigation PID controllers

There are of course many parameters one may configure using the serial communication, all the native commands of the quad rotor helicopter can be found in [1].

Figure 23 is a screen capture of the configuration mode of the GCS in the visual view. It illustrates how the raw sensor data are converted to visual objects for better user interface. Furthermore, all the data flowing on the serial line can be viewed as logs in the "data console" tab of the configurator.

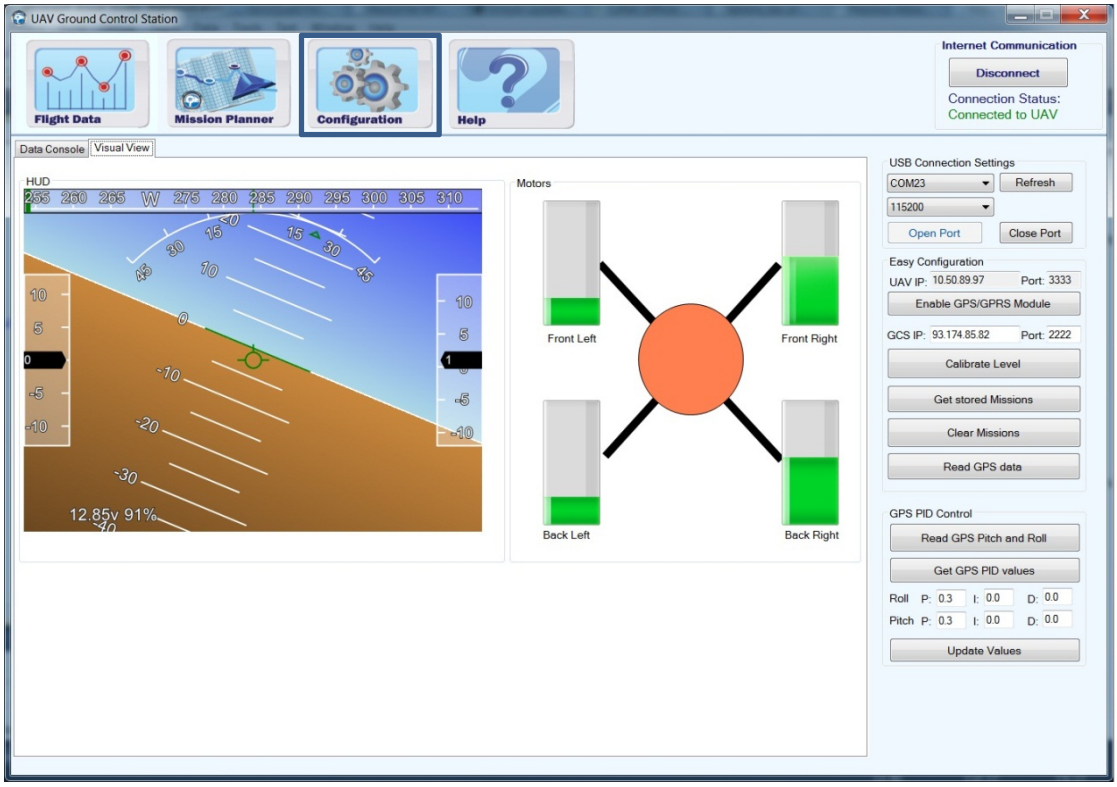


Figure 23: The Configuration View of the GCS

4.3.3 The Mission Planner View

The Mission planner view of the GCS is the environment the user can plan the flight of the UAV. As shown in Figure 24, the user can easily select the positions on the globe for surveillance. These positions are appended to the mission list appear on the right hand side. The user has therefore the opportunity to upload them to the UAV either using the wired or wireless communication. The mission list can also be saved for future upload or mission repeat.

For the development of mission planner, the Google Maps .NET library used [24] as it features useful functions for extracting and importing information to map. It also provides functions for converting mouse position to WGS84 coordinates and vice versa. The mission planner uses both internet communication and serial communication handlers to exchange data with the UAV.

Last but not least, "find location" functionality added. This helps the user to search a country, a town or even a street address for easier mission planning procedure. The altitude mode, which is shown as disabled in Figure 24, is a functionality of the mission planner to receive altitude information provided by Google Maps. This can be used for future work on the project for altitude prediction. More details on this can be found in chapter 7.

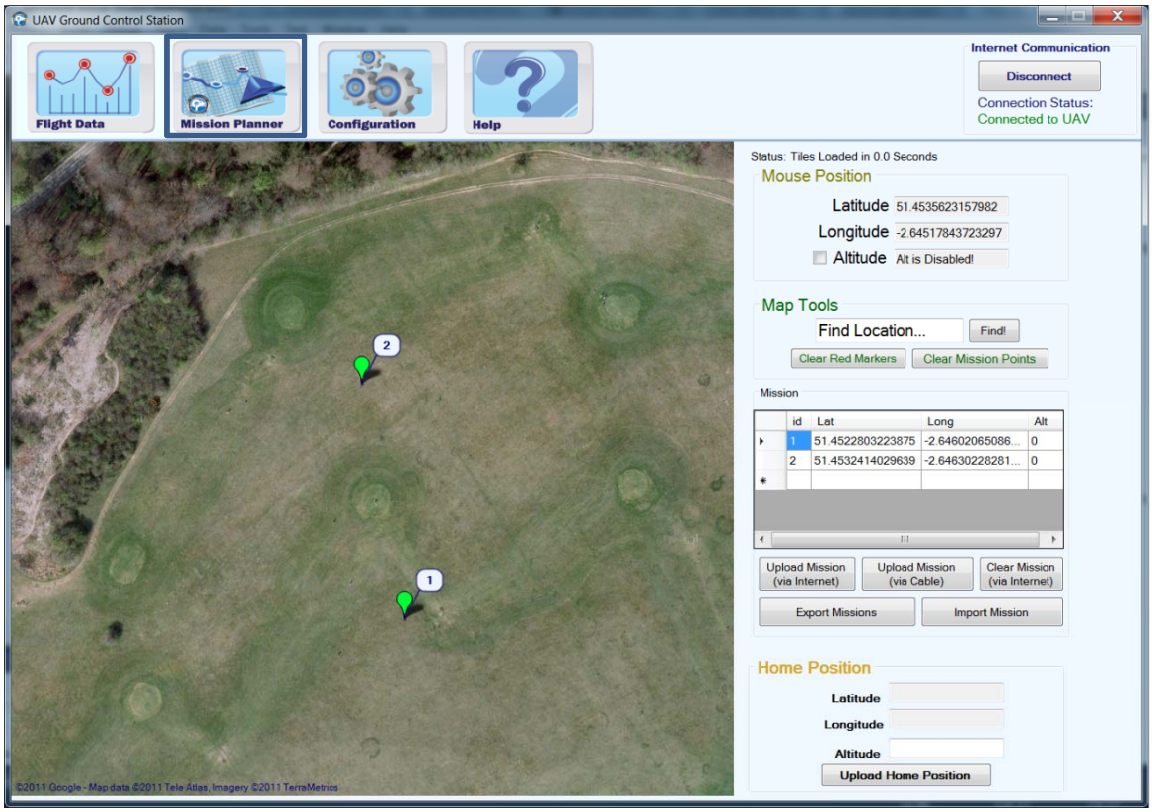


Figure 24: Mission planner view of the GCS

4.3.4 The Flight Data View

This is the virtual cockpit of the user. It uses the internet connection handler to obtain the recent flight information coming from the UAV and illustrates them. As shown in Figure 25 the vehicle is tracked on the map. The current flight data are shown in the HUD and GPS details box placed on the left hand. The information shown are the GPS information, the current roll and pitch angles as well as the altitude and magnetic heading.

Furthermore, through this view, the mission amend objective is achieved. The user by right clicks a position on the map and then presses "Fly here", a message is generated instructing the UAV to change it trajectory. Finally, the flight data view keeps logs of the flight in the "Remote data" tab shown under the HUD. These data can be saved for further analysis.

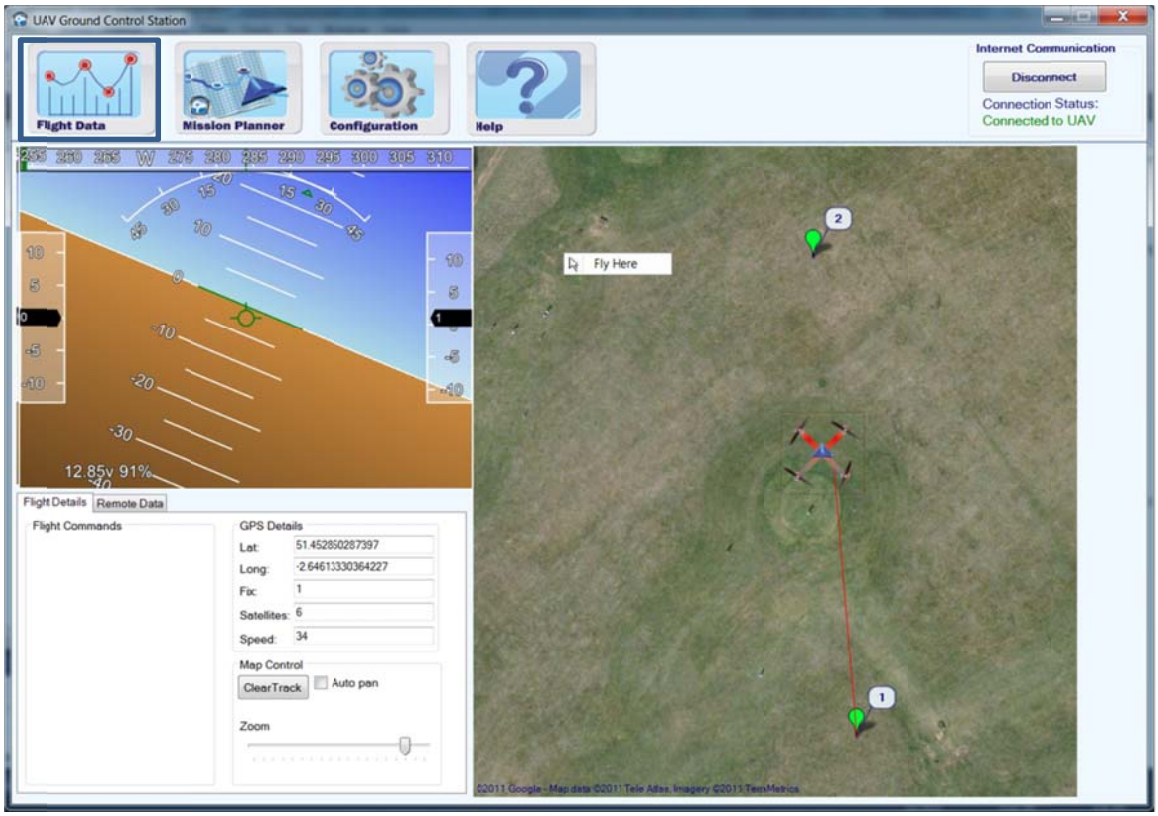


Figure 25: Flight data view of the GCS

5. RESULTS

This chapter is dedicated to the results obtained during the verification of the UAV's system functionality. The following components were tested:

- Main loop execution time of the UAV
- Autonomous navigation algorithm
- Communication link (both UAV and GCS)
- GCS functionality

5.1. Testing the Control loop rate

One of the very first tests were made was the observation of the control loop rate. By introducing the new features described in the "UAV Software", found in chapter 4.2, there was a risk that the small – 16MHz microcontroller might not be able to handle the processing power requirements.

However, as shown in Figure 26 the results obtained prove that there was not any noticeable effect on the loop rate in a way to affect the main loop. On average, the loop time was increased by just 20 microseconds with some peaks of 300 microseconds when there was data transmission from GCS to the UAV. Nevertheless, that is not worrying because such transmissions are performed before the aircraft takes-off and rarely during flight when a mission is amended.

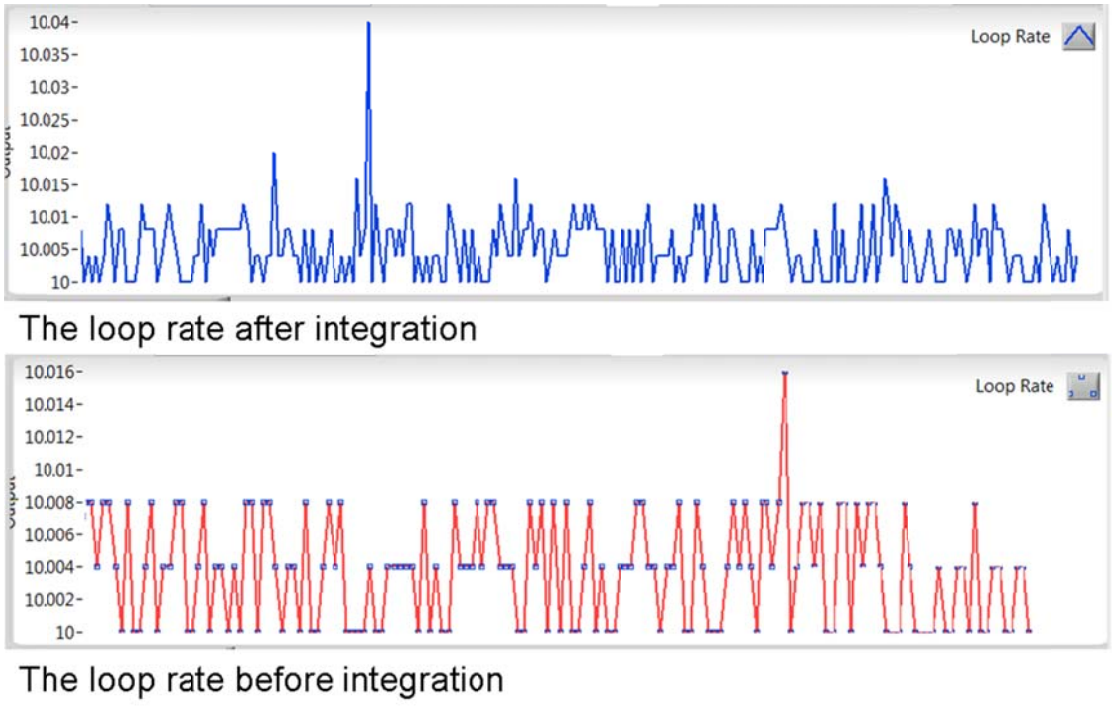


Figure 26: The Control loop rate before and after the UAV software integration

5.2. Autonomous Navigation Algorithm Test

This chapter analyses the results of real tests made in an open area for the autonomous navigation algorithm verification, however the propellers were not installed for safety reasons. The first test made was to check if the autonomous navigation algorithm can generate the correct motor commands. For simplicity, the north is assumed as the vehicle's front projection. Using the GCS, a position to the east (with respect to vehicle front) was loaded as mission. As mentioned before, the UAV requires manual take-off before it can start its autonomous navigation process. Hence, using the transmitter, a throttle value was set. Figure 27 was captured just after the throttle value was adjusted. It shows that all the motors are spinning at the same speed. This is what expected because the throttle attempts to give the vehicle only the vertical movement. As soon as the propellers were taken off, what is shown is that the vehicle hovers at level position.

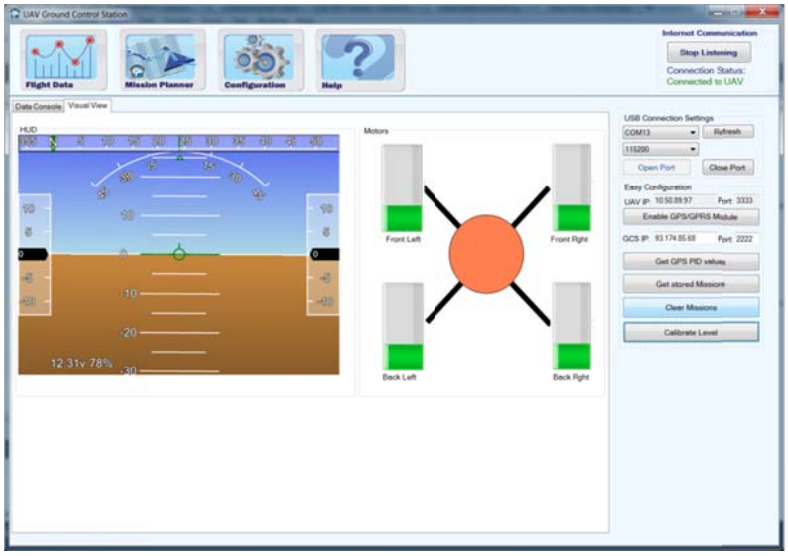


Figure 27: The GPS Navigation algorithm is disabled

Figure 28 was captured just after the autonomous navigation algorithm enabled. As soon as eastern destination was set, the motor speeds of "front-left" and "back-left" were increased while the "front-right" and "back-right" speeds were decreased. That indicates that the algorithm has correctly calculated the roll and pitch angles. The motor speeds shown in the figure, attempt to provide a clockwise roll angle to the vehicle to reach its destination.

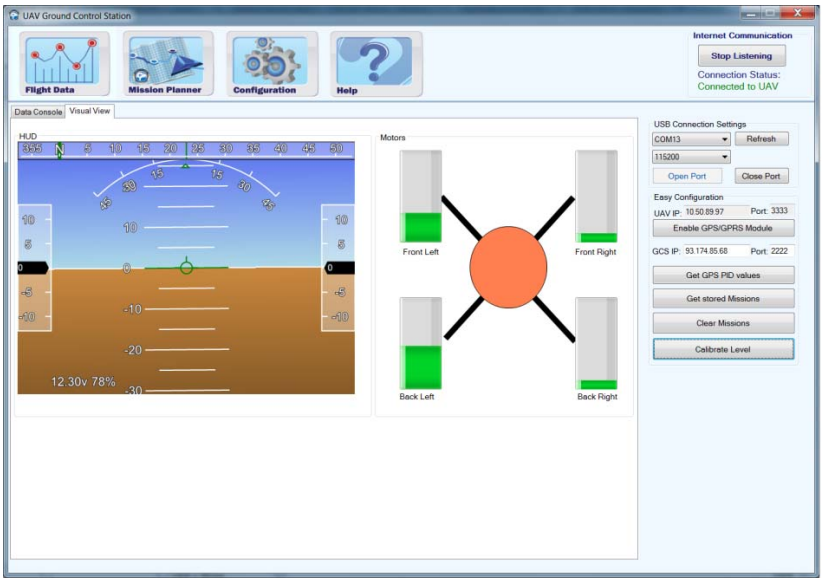


Figure 28: The GPS Navigation algorithm is enabled

Thus, the next test made was to bring the vehicle to the destination position and observe its response. This test was made to verify the position hold functionality of the algorithm as well as the determination of a reached position. When the vehicle placed to the destination, the response on the motors was remarkable. The motors "front-left" and "back-left" decreased their speeds. However, all the motors were not spinning at the same speed. That is because the inaccuracy of the GPS. For the confirmation that the GPS inaccuracy has resulted this response, the vehicle moved eastern than the destination position. The results obtained were as expected. The "front-left" and "back-left" motors decreased their speeds while the "front-right" and "back-right" motor speeds were increased to bring it back to the destination. A drawback recognised is the low GPS rate. As soon as the navigation between positions operates correctly (which includes the magnetometer readings for the trajectory following), the main reason the destination is delayed to marked as reached is the GPS rate.

Autonomous navigation algorithm result synopsis:

- The autonomous navigation algorithm has proven to function correct as soon as it calculates the required roll and pitch angles
- The position hold feature proven to work as it can hold the vehicle in a radius of about 15 meters to destination
- There is lag of the algorithm for detecting immediately that the destination is reached. That is due to the low GPS refresh rate.

5.3. Communication Link Tests

The way the communication link verified was by testing each side individually for both inbound and outbound data transfer. At first, an infinite loop was created on the microcontroller board that was generating random data. The data was in the form of messages the GCS understands and presents. At the same time, the microcontroller was connected to the computer via the serial port (USB). The messages were sent through the internet (outbound network socket) connection were repeated on the serial communication link. A telnet client as well a serial line client was used to log both incoming transmissions. After certain time has elapsed, both log files were compared for any data misses or corrupts assuming as valid data the data received through the wired transmission. The results showed that every byte was sent over the internet were received correctly and in order.

For the inbound transmission verification, a similar concept applied. The difference was that the infinite loop on the microcontroller used to read from the internet link (network socket) and then repeat them to the serial line. The data received from the serial line were compared with the original data was sent from the computer's outbound internet connection. Again there were not any corruptions or data misses. Identical procedure applied to verify the GCS communication link.

After the individual verification tests were completed, the communication link was tested while the autonomous navigation algorithm was running. The serial communication kept active to validate the transferred mission plans and mission amendments. Succeed transmission of data was resulted.

However, tests showed that the GSM antenna interferes with a component is installed on the PCB. For this reason the antenna moved as far as possible from the electronics.

5.4. GCS Functionality Tests

Once the GCS developed the functionality tests taken place was through the internet and serial communication lines the software relies on. A telnet client was used to imitate the UAV in terms of data transmission through internet. Random data were sent and the transmission rate was faster than the GPRS usually handles. That ensured the memory allocated successful as there was not failures after certain time has elapsed. That also ensures that the GCS can be used with a UAV transmits over a faster communication link.

Furthermore, the GCS tested for any possible failures on invalid data transmissions. The GCS proven to reject any messages are not formatted as it could. Finally the GCS verified that generates messages which the UAV can understand and decode as discussed in sub-chapter 0.

6. EVALUATION

The project had three main objectives. These were the integration of the remote controlled quad rotor helicopter to an autonomous vehicle, the generation of the GCS software as well as the communication link between them.

The integration of the software to a UAV objective was achieved, as soon as the UAV can travel through different positions and hold its position (hovers) for surveillance. However, automatic take-off and landing procedures are not yet functional and a RC transmitter is still required. There were two main reasons affected this objective. The first was the minimal time remained for software developing after the hardware assembly of the aircraft. Apart from that, the PCB was defective. A lot of time spent for the determination of whether was a PCB design error or due to a hardware component failure. Finally, the problem proven to be a manufacture error as there was resistance in the thru-hole vias.

On the other hand, the GCS application developed achieved all of its objectives. These are the three operational views have been described in chapter 4.3 (configuration, mission planning and flight data representation).

Furthermore, the communication link objective achieved as there is a bidirectional communication between the UAV and the GCS. The drawback of the communication link is its speed, but it was the only available solution featured long-range communication with minimal size and weight. In addition, GSM, which the GPRS relies on, is widely available in inhabitant areas where the project targets on as application environment.

7. FUTURE WORK

Regarding the improvements would be made on the current project; the GPS receiver could be replaced with one that offering better update rates. Such receivers are now available at low cost. Thus, the issue of delayed determination of a reached position will be solved.

Furthermore, a new version of the remote controlled quad rotor software has been released. According to the description, the "altitude hold" algorithm became reliable. Hence, the modifications have been made on this project can be transferred to the new version of the software. As a result, the automatic take-off and landing functionality can be integrated there.

On the other hand, the GCS software can be improved by an algorithm predicts the trajectory of the UAV on the globe. As soon as the GCS software already obtains altitude information for a specified position, the trajectory prediction algorithm can be used to inform the UAV to change its altitude to avoid obstacles.

A worthy improvement of the project could be the replacement of the 16MHz microcontroller with a faster microprocessor or a digital signal processor. Such improvement will increase the stability of flight. Moreover, new functions that require further processing resources can be implemented.

8. BIBLIOGRAPHY

[1] "Aeroquad, The Open Source Quadcopter," [Online]. Available: http://www.aeroquad.com.

[2] E. Pastor, J. Lopez and P. Royo, "UAV Payload and Mission Control Hardware/Software Architecture," *Aerospace and Electronic Systems Magazine, IEEE,* vol. 22, pp. 3-8, June 2007.

[3] S. Park, D. Won, M. Kang, T. Kim, H. Lee and S. Kwon, "RIC (robust internal-loop compensator) based flight control of a quad-rotor type UAV," *IEEE/RSJ International Conference onIntelligent Robots and Systems, 2005. (IROS 2005).,* pp. 3542- 3547, 2005.

[4] A. Tayebi and S. McGilvray, "Attitude stabilization of a VTOL quadrotor aircraft," *IEEE Trans. Control Systems Technology,* vol. 14, no. 3, pp. 562 - 571 , 2006.

[5] teipir, "PID Controllers," [Online]. Available: http://gun.teipir.gr/DSAELAB/Ergastiriakes/pidtutorial.pdf.

[6] Wikipedia, "PID controller," 2011. [Online]. Available: http://en.wikipedia.org/wiki/PID_controller.

[7] N. Metni, J.-M. Pflimlin, T. Hamel and P. Soueres, "Attitude and gyro bias estimation for a flying UAV," *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on ,* pp. 1114 - 1120 , 2005.

[8] P. Martin and E. Salaun, "The true role of accelerometer feedback in quadrotor control," *IEEE International Conference Robotics and Automation (ICRA),* pp. 1623-1629, 3-7 May 2010, 2010.

[9] W. Premerlani and P. Bizard, "Direction Cosine Matrix IMU: Theory," [Online]. Available: http://gentlenav.googlecode.com/files/DCMDraft2.pdf.

[10] "NMEA Description," [Online]. Available: http://www.cs.put.poznan.pl/wswitala/download/pdf/NMEAdescription.pdf.

[11] G. Baddeley, "GPS - NMEA sentence information," 2001. [Online]. Available: http://aprs.gids.nl/nmea/.

[12] S. Hrabar, "3D path planning and stereo-based obstacle avoidance for rotorcraft UAVs," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008.

[13] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007.

[14] M. Barr, "Embedded Systems Programming," in *Pulse Width Modulation*, 2001, pp. 103-104.

[15] "I2C Manual," Philips Semiconductors, [Online]. Available: http://www.nxp.com/documents/application_note/AN10216.pdf.

[16] A. Soumelidis, *Design of an embedded microcomputer based mini quadrotor UAV,* unpublished.

[17] J. Cai and D. Goodman, "General packet radio service in GSM," *Communications Magazine,* vol. 35, no. 10, pp. 122-131, 1997.

[18] M. Jovanovic and D. Starcevic, "Software Architecture for Ground Control Station for Unmanned Aerial Vehicle," in *Computer Modeling and Simulation*, 2008.

[19] I. Papini, "Vehicle Simulator reference manual," 20 July 2010. [Online]. Available: http://www.hangsim.com/vsf/help/print.htm.

[20] A. Corporation, "ATmega2560," [Online]. Available: http://www.atmel.com/dyn/products/product_card.asp?category_id=163&family_id=607&subfamily_id=1723&part_id=3632.

[21] Arduino, "Arduino Mega," [Online]. Available: http://arduino.cc/en/Main/ArduinoBoardMega.

[22] J. Zhong, "PID Controller Tuning: A Short Tutorial," 2006. [Online]. Available: http://saba.kntu.ac.ir/eecd/pcl/download/PIDtutorial.pdf.

[23] "AT Commands Reference Guide," Telit Wireless solutions, [Online]. Available: http://www.telit.com/module/infopool/download.php?id=542.

[24] radioman, "GMap.NET - Great Maps for Windows Forms & Presentation," 2011. [Online]. Available: http://greatmaps.codeplex.com/.

[25] T. Madani and A. Benallegue, “Backstepping Control for a Quadrotor Helicopter,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3255-3260, 2006.

[26] G. Mao, S. Drake and B. Anderson, “Design of an Extended Kalman Filter for UAV Localization,” *Information, Decision and Control, 2007. IDC '07* , pp. 224 - 229 , 2007.

[27] S. Bouabdallah, A. Noth and R. Siegwart, “PID vs LQ control techniques applied to an indoor micro quadrotor,” in *Intelligent Robots and Systems*, 2004.

9. APPENDICES

9.1. UAV SYSTEM USER MANUAL

!!!WARNING!!!
Be aware that the propellers attached on the motors are really dangerous components. The motors can obtain speeds of 11,665 revolutions per minute !!!
Please remove the propellers before proceeding to step 1

STEP 1 - Loading the sketch

1. **If you are planning to use GPRS connectivity, please attach a SIM card in the slot**
2. **Using the Arduino IDE software, load the sketch named: QuadrotorUAVsoftware.pde**
3. **Locate the file named Aeroquad.h**
4. **Find the line: #define SIMPIN ""**
5. **Enter the pin code of the SIM card if any (leave it as is if you have not set any PIN code on the card)**
6. **Locate the file: GM862.cpp**
7. **Find the line contains**
8. **Modify the following line according to the SIM network you are on:**
   requestModem("AT+CGDCONT=1,\"IP\",\"m-bb.o2.co.uk\"", 3000, true, buf);
   (leave it as is if you are on O2 mobile broadband)
9. **Save and upload the sketch**

STEP 2 – Calibration

1. **Visit the www.aeroquad.info to download the configurator software (note: the software transformed to UAV is 2.4, find a compatible version of configurator).**
2. **Connect the USB cable and start the configurator, set as COM the one the Arduino is connected to.**
   Baud rate of COM port: 115200
3. **Follow the instructions to calibrate the EEPROM, ESCs RC transmitter, and the magnetometer**
4. **Close port and then close the configurator Software**

STEP 3 – GPS Navigation setup

1. **Start the UAV GCS software**
2. **Select the COM port Arduino is connected, default baud rate (115200) and click "Open Port"**
3. **Place to vehicle to a level position and navigate to "Visual View" tab. Verify the vehicle appears level. If not click the "calibrate level" button**
4. **Click the "Get GPS PID values" if you have not changed the default ones, ensure they are: 0.3;0;0;0.3;0;0**
5. **Ensure your firewall does not block the GCS port shown. If you are behind a router, forward that port on your computer.**
6. **Click the "Start listening" button found on top right position**
7. **Click the "Enable GPS/GPRS module". Wait until the UAV reports "ready!" on the console. The "UAV IP:" will become available. The connection status will change to "Connected to UAV"**

STEP 4 – Plan your missions

1. **You can disconnect the USB cable now unless you prefer uploading the mission via cable**
2. **Navigate to the "Mission planner" view**
   NOTE: You can upload up to 10 positions max!
3. **Find the places you are interested of and mark them up:**
   Right clicks are sets new marker
   Left click on marker removes it
   (Note: only green markers are transferred to the vehicle for navigation)
4. **Upload the mission either via the USB cable or via the internet link**
5. **Optional: navigate back to configurator mode and click "Get stored missions". Verify the correctness.**

STEP 5 – The flight

1. **Navigate to the "flight data" view**
2. **Navigate to "Remote Data" tab, you will see raw data coming from the UAV and the HUD will start moving**
3. **Wait until the "GPS fix" led indicator on the UAV is turned on.**
4. **Test that the "GPS Nav" LED is getting ON when switching the upper right auxiliary button of the transmitter.**
5. **Turn it off**
6. **Take-off the vehicle to the desired altitude.**
7. **Enable the autonomous navigation by turning the upper right auxiliary button.**

The "Flight data" view keeps track of the UAV. You can also observe its orientation. Check the battery voltage-percentage on a regular basis.

To change its trajectory, just right click the map to the desired position and click "fly here"

If the last position in the mission list is the base, it will hover above you. If not, use the "fly here" feature to bring it near you.

Land it by turning off the GPS Navigation and using the throttle stick.

9.2. PCB Hardware Design - Interface

The following figures show the design of the PCB developed. On the right hand side the schematic is shown while on the left is the actual board magnified