Measurement 119 (2018) 142–155

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

Drone as an autonomous aerial sensor system for motion planning

Gavin J. Fouché, Reza Malekian*

Department of Electrical, Electronic and Computer Engineering, University of Pretoria, Pretoria 0002, South Africa

ARTICLE INFO

Keywords:
Unmanned Aerial Vehicle
Aerial sensor system
Autonomous navigation
Motion planning

ABSTRACT

A system capable of both autonomous navigation and remote fire detection was developed from first principles. Using a complementary filter and attitude fundamentals, an Euler coordinate system representation of the aircraft's orientation was measured using a widely available low-cost inertial measurement unit incorporating MEMS accelerometers, gyroscopes and magnetometers. Using line-of-sight guidance principles, navigation trajectories could be calculated in real-time providing autonomous navigation between user designated waypoints. A stabilisation control system using proportional-integral-derivative (PID) controllers was developed in order to achieve stabilised flight on the calculated navigation trajectories. Fire detection was facilitated with the use of three low-cost air composition sensors fed into an artificial neural network. A motion planner was developed to assist in flight planning by using topology information for the flight area to construct an optimal flight path based on distance and climb penalties.

1. Introduction

Unmanned Aerial Vehicles (including drones) have become a prominent solution to many problem areas in recent times due to advancements in lithium-based batteries, microprocessor technology and inertial measurement sensors for stabilisation. The smart-phone industry helped to drive down the cost of these technologies drastically and in a four-year period the cost of micro-electromechanical gyroscopes fell by a factor of 10 [1]. The success of UAVs (unmanned aerial vehicles) in military applications and the reduced cost advantages have made them more appealing to the private sector.

The terrain of Southern Africa is mountainous and unforgiving in many regions. This, coupled with low population density, produces a setting where land survey missions are few and often over large areas. For this reason, fixed-wing drones have become popular in this role, such as in anti-poaching operations [2]. Despite our growing understanding of fires and the technological advances available to prevent them, the number of fires in South Africa has stayed a consistent average of 332,000 every year for the last ten years, with over ZAR 4,4 million worth of damage due to forests, agriculture and grassland fires and over ZAR 117 million damage in informal areas done alone in 2013 [3].

In navigation and path planning, several approaches exist for line-of-sight navigation [4]. Most autopilot systems use an AHRS (Attitude Heading Reference System) which outputs pitch, roll and yaw—derived from a multitude of inertial measurement sensors including accelerometers, gyroscopes and magnetometers to produce an expression of the model's orientation [5]. When these inertial measurement sensors are combined, they form an IMU (inertial measurement unit). This AHRS is coupled with GPS (global positioning system) information to provide combined location and attitude information in what is known as an Inertial Navigation System (INS) which provides navigational instructions [6].

Attitude estimation was achieved by developing a sensor fusion approach to fuse accelerometer, gyroscope and magnetometer data using Euler orientation fundamentals, flight force compensation, magnetic interference correction, tilt compensation and filtering in the form of a digital complementary filter approach based on Euler angles. In sensor fusion, Kalman filters can provide an excellent state estimation for orientation by estimating gyroscope bias error and filtering out accelerometer noise, but their accuracy leads to increased computational complexity. By comparison, complementary filter approaches using a combination of low-pass and high-pass filters provide an almost equally adequate solution if they implement adequate centripetal force compensation models [7]. In small drone applications the complementary filter has been proven to show similar robustness with significantly less computational cost [8,9].

Various implementations and approaches to fire detection and recognition exist currently, using optical colour cameras similar to webcams [10,11], as well as smoke density and temperature sensors [12]. Image-based approaches have shown formidable detection accuracy rates above 70% but incur significantly more computational overhead for image processing. The proposed fire detection method would be developed using conventional air composition and temperature sensors

* Corresponding author.
E-mail address: reza.malekian@up.ac.za (R. Malekian).

https://doi.org/10.1016/j.measurement.2018.01.027
Received 14 September 2017; Received in revised form 15 January 2018; Accepted 17 January 2018
Available online 01 February 2018

on the craft due to their small weight, price and associated computational power required in comparison to traditional camera-based approaches. The combination of these off-the-shelf sensors in conjunction with artificial neural networks has been proven to work effectively as a fire detection method [13].

The first engineering challenge was the Attitude Heading Reference System (AHRS). Attitude estimation was achieved by developing a sensor fusion approach to fuse accelerometer, gyroscope and magnetometer data using Euler orientation fundamentals, flight force compensation, magnetic interference correction, tilt compensation and filtering in the form of a digital complementary filter approach based on Euler angles. In sensor fusion, Kalman filters can provide an excellent state estimation for orientation by estimating gyroscope bias error and filtering out accelerometer noise, but their accuracy leads to increased computational complexity. By comparison, complementary filter approaches using a combination of low-pass and high-pass filters provide an almost equally adequate solution if they implement adequate centripetal force compensation models [7]. In small drone applications the complementary filter has been proven to show similar robustness with significantly less computational cost [8,9].

The inertial navigation system which was developed using inertial navigation system fundamentals with GPS and the implemented attitude heading reference system. Guidance principles were developed using missile line-of-sight guidance principles using equirectangular projection. Altitude and airspeed estimation were developed using fundamental physics principles from on-board ambient and dynamic pressure sensors. Equirectangular projection provides an effective way of localising a small area with minimal error. It reduces the problem to trigonometry and by assuming a fixed Earth radius and utilising a constant origin angle ($\cos(\phi_0)$), it provides one of the least processor intensive methods to be implemented on a micro-controller [14].

For the flight stabilisation control system, Digital PID controllers were developed to stabilise the aircraft on the fundamental axes required for autonomous flight. A three-dimensional characteristic model was developed to simulate the chosen aircraft using *Laminar Research X-Plane Plane Maker* which was based on similar electric aircraft models. Control characteristics of the aircraft were determined and the PID controllers were manually tuned and tested in simulation using a desktop autopilot simulator developed in *C#* that communicates with the *Laminar Research X-Plane* flight simulator. The PID constants obtained through simulation were then implemented on the micro-controller.

Due to the high cost in weight, computational power and price introduced by camera based image processing detection methods. The proposed fire detection method will be developed using conventional air composition and temperature sensors on the craft due to their small weight in comparison to traditional infrared cameras. The combination of these off-the-shelf sensors in conjunction with artificial neural networks has been proven to work effectively as a fire detection method [13].

Path-planning algorithms were developed by utilising equirectangular projection, terrain meshes and artificial intelligence techniques to autonomously find an optimal flight plan to minimise travel distance and maximise efficiency. Equirectangular projection was used to linearise the geographic coordinate system. Terrain meshes were constructed using the NASA SRTM terrain database and the A^* graph search algorithm was implemented to find the optimal path using a heuristic function based on distance and energy loss.

This paper is organised as follows. Section 2 describes the system with a short overview highlighting key functionality of the architecture components. Section 3 notes the key design of the proposed system with a focus on sensor fusion. Findings and results are summarised and illustrated in Section 4. Section 5 proposes some possible future work and improvements over the proposed system. Section 6 concludes the paper.

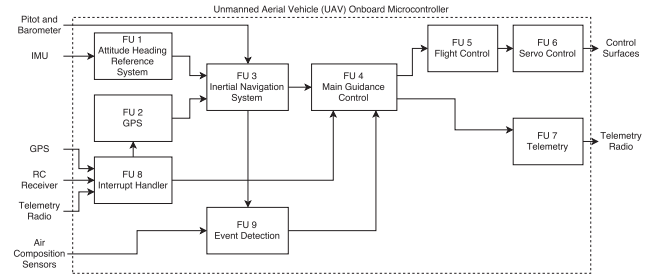


Fig. 1. Functional block diagram of UAV software components.

2. System overview

The broad system overview is shown in Fig. 1. All system components are split into discrete functional units responsible for independent operations relating to the system.

The attitude heading reference system (functional unit 1) calculates the attitude orientation of the aircraft using data obtained from an on-board mounted inertial measurement unit. In aircraft navigation, the attitude is the airframe local coordinate system used to describe an aircraft's orientation in three-dimensional space. This orientation coordinate system is with reference to Earth, and in essence describes how many degrees each axis is orientated away from the Earth's horizon. It is therefore implied that the inertial measurement unit should be capable of estimating the orientation of the airframe with respect to Earth. An Euler angle coordinate system was chosen over the quaternion system despite the known singularity error (gimbal lock) [14]. This decision was taken since the aircraft does not intend performing aerobatic manoeuvres nearing ninety degrees of bank in any given axis (where the singularity occurs). Euler angles do not require any complex transformations as in the case with quaternions, thus making the implementation less computationally intensive.

The GPS software component receives standard National Marine Electronics Association (NMEA) formatted messages from the interrupt handler (FU 8 in Fig. 1). The primary purpose of the GPS component is simply to parse these serial messages. Several NMEA messaging formats exist to relay different sets of information. For this application the most relevant are the GPGGA and GPRMC information messages that contain information on the current time, speed, position, course, magnetic variation and altitude. Once parsed, this information is used by the Inertial Navigation System (FU 3 in Fig. 1).

The Inertial Navigation System (INS, FU 3 in Fig. 1) is responsible for using information from both the Attitude Heading Reference System (FU 1) as well as the GPS (FU 2) in order to establish where the aircraft is in world space as well as how it is moving with respect to its orientation. Altitude can be estimated with the use of a barometer and temperature sensor. One such sensor is the BMP180. The barometer measures ambient pressure, and when used in conjunction with a temperature sensor it can provide reliable altitude information. GPS altitude is accurate but slow to react, whereas barometric altitude is fast to react but suffers from some offset error due to atmospheric pressure that varies between regions and climates. The barometric altitude must thus be corrected by GPS altitude. Combining both the AHRS and GPS information provides a more robust estimation as to the whereabouts and overall state of the system. This includes filtering and combining multiple variables from both the GPS and AHRS respectively that would in isolation be unreliable. GPS information is slow but otherwise highly accurate whereas the AHRS information is fast but noisy. A combination of the two into one single unified Inertial Navigation System provides a more appropriate solution that can be used in conjunction with guidance control to navigate the aircraft between designated waypoints in an adequate fashion. The INS calculations for airspeed and altitude are facilitated by a differential pressure sensor (MPXV7002DP) and barometric sensor (BMP180) respectively.

The main guidance control unit (FU 4 in Fig. 1) tracks the current flight mode by interpreting the radio transmitter flight mode selection through the interrupt handler. It also monitors information from the event detection system and Inertial Navigation System to provide situational awareness and informational prompts to the ground control station via telemetry radio. Using information from the Inertial Navigation System it makes informed decisions on how to guide the aircraft to its mission waypoints by passing the necessary commands to the flight control unit. These commands will typically consist of information on what orientation to maintain in order to reach the next desired waypoint.

The flight control unit (FU 5 in Fig. 1) is responsible for using control systems to stabilise the aircraft in flight. Flight control parameters such as pitch, roll, heading and altitude from the main guidance control unit (FU 4) are interpreted by the flight control unit and the necessary control outputs are driven to the servo control unit (FU 6) which in turn moves the flight control surfaces of the aircraft to maintain the set orientation. This unit therefore serves as the primary constituent of the autopilot system.

The servo control software component (FU 6 in Fig. 1) is responsible for providing the functionality to drive the control surface actuator servos. It receives control commands from the flight control unit (FU 5) and generates an adequate pulse-width modulated output for each channel to manipulate the servo arm for each respective control surface. The servo control unit uses timers to achieve pulse-width modulated signals.

The telemetry software component (FU 7 in Fig. 1) facilitates communication with the ground control station, primarily outputting serial messages from the main guidance control unit (FU 4) via radio telemetry to the ground station.

The primary purpose of the interrupt handler software component (FU 8 in Fig. 1) is to handle the hardware interrupts coming from a number of different stimuli. This includes receiving serial GPS updates, measuring PWM signals from the RC receiver and capturing serial commands from the telemetry radio. Once captured, this information is recorded and kept for use in the main guidance control unit (FU 4).

The event detection software component (FU 9 in Fig. 1) contains all functionality for the detection of fires, most notably the use of an artificial neural network to classify fires based on the analog air composition sensor data.

3. System design

3.1. Attitude heading reference system

The basic flow of operations for the proposed attitude heading reference system is illustrated in Fig. 2 below.

The mathematical expressions can be derived using the Euler angle coordinate system for representing a motionless rigid body in the global reference as shown in Fig. 3. The Euler body frame reference system with the origin at aircraft centre of gravity with x, y and z components has the angular components pitch (θ), roll (ϕ) and azimuth yaw (ψ).

The measured acceleration vector on the aircraft body is henceforth denoted $\overrightarrow{a_m}$. The measured angular rate vector is denoted $\vec{\omega}$ and the magnetic vector is denoted $\vec{m}$. All 3-axis measurements have components x,y and z orientated with the Euler convention as in Fig. 3.

In order to estimate the attitude of the aircraft, knowledge of the true gravity vector $\overrightarrow{a_g}$ is required. As shown in Eq. (1) and (2), from the

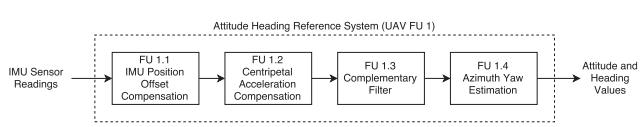


Fig. 2. Functional block diagram of attitude heading reference system constituent components.

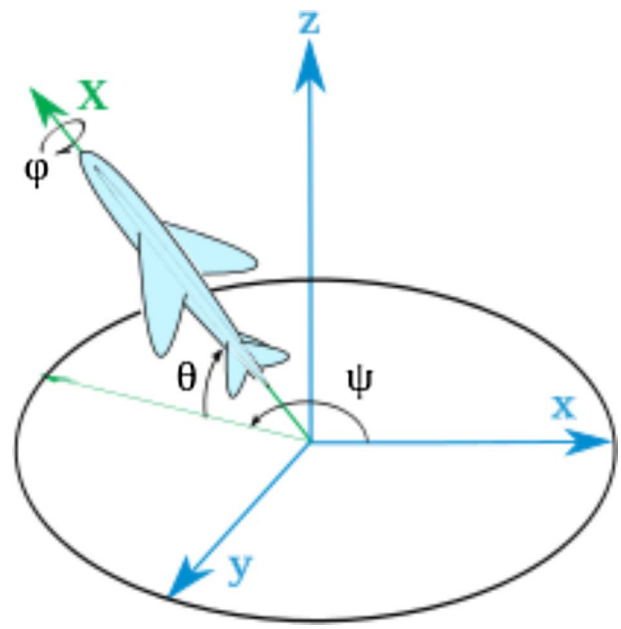


Fig. 3. Euler coordinate system components.

true gravity vector we can calculate the pitch and roll angles. The azimuth yaw angle (or heading) can be calculated from the magnetic vector as in Eq. (3).

$$\theta = \text{arctan2}(a_{gx}, a_{gz}) \tag{1}$$

$$\phi = \text{arctan2}(a_{gy}, a_{gz}) \tag{2}$$

$$\psi = \text{arctan2}(m_x, m_y) \tag{3}$$

where the function arctan2(y,x) is defined out of convenience as:

$$\text{arctan2}(y,x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geqslant 0, \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0, \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0. \end{cases} \tag{4}$$

Due to the fact that the IMU is not perfectly aligned with the aircraft centre of gravity (as illustrated in Fig. 4), a transformation must be performed to estimate the measurements at the centre of gravity as opposed to those measured at the IMU position. From Eq. A-5 in [15] it follows that the acceleration correction can be obtained by Eq. (5) below. Where $\overrightarrow{a_m}$ is the corrected measured acceleration at centre of

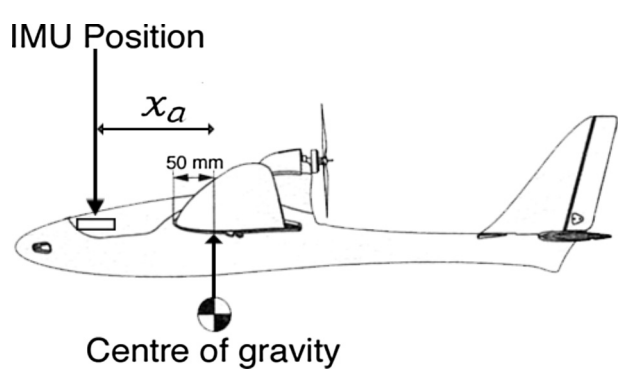


Fig. 4. Illustration of IMU mount position vs the position of the centre of gravity.

gravity, $\vec{a}_{IMU}$ is the measured acceleration at the IMU, $\overrightarrow{A}$ is the positional offset compensation vector (as in Eq. (7)) and x_a,y_a and z_a are the IMU positional offsets from the location of the centre of gravity.

$$\vec{a}_m = \vec{a}_{IMU} + \overrightarrow{A}\begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} \tag{5}$$

$$\overrightarrow{A} = \begin{bmatrix} \omega_x^2 + \omega_z^2 & -(\omega_x\omega_y - \dot{\omega}_z) & -(\omega_x\omega_z + \dot{\omega}_y) \\ -(\omega_x\omega_y + \dot{\omega}_z) & \omega_x^2 + \omega_z^2 & -(\omega_y\omega_z - \dot{\omega}_x) \\ -(\omega_x\omega_z - \dot{\omega}_y) & -(\omega_y\omega_z + \dot{\omega}_x) & \omega_x^2 + \omega_y^2 \end{bmatrix} \tag{6}$$

If the IMU is mounted in the forward centre section of the fuselage, far enough to isolate it from the disturbance of the electric motors magnetic field, it can be assumed that $y_a \approx 0$ and $z_a \approx 0$. Furthermore, in assuming $\dot{\vec{\omega}} \approx 0$, Eq. (5) can thus be simplified to:

$$a_{mx} = a_{IMUx} + (\omega_x^2 + \omega_z^2)x_a \tag{7}$$

$$a_{my} = a_{IMUy} - (\omega_x\omega_y)x_a \tag{8}$$

$$a_{mz} = a_{IMUz} - (\omega_x\omega_z))x_a \tag{9}$$

After the IMU position offset correction, the centripetal acceleration estimation model proposed in [7] is used to correct for centrifugal force effects. The acceleration due to centripetal force ($\vec{a_c}$) is expressed as a function of body velocity () and gyro angular rates ($\vec{\omega}$) in Eq. (10).

$$\vec{a_c} = \vec{\omega} \times \vec{V}_{body} \tag{10}$$

The true gravity vector $\vec{a_g}$ is estimated by subtracting the centripetal acceleration from the measured acceleration as in Eq. (11). This provides the crucial true gravity acceleration needed to compute Euler angles.

$$\vec{a_g} = \vec{a_m} - \vec{a_c} \tag{11}$$

Due to the fact that only the longitudinal axis body velocity can be measured with a pitot tube and as there is no additional instrumentation to adequately measure angle-of-attack as used in the centripetal compensation model [7], all body velocity is assumed in the longitudinal axis, so for the proposed compensation model approach:

$$\vec{V}_{body} = [V_{air} \quad 0 \quad 0] \tag{12}$$

This longitudinal velocity is assumed as airspeed V_{air} as measured by the airspeed estimation component of the Inertial Navigation System (UAV FU 3) to be discussed in subsequent sections.

Due to the high frequency noise characteristics present in the accelerometer which are not easily quantifiable (such as those introduced by vibration) a complementary filter approach is proposed. Using measured gyroscope angular rates which are not susceptible to similar disturbances, the complementary filter can estimate attitude based on the gravity vector as well as the angular rate.

Using the fundamental assumptions in Eq. (13) and (14) as well as the use of the true gravity vector and arctan2 function as in Eqs. (1) and (2) the complementary filter approach is illustrated in Fig. 5.

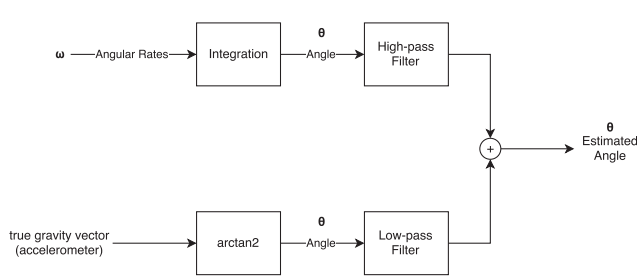


Fig. 5. Block diagram showing complementary filter structure.

$$\theta \approx \int (\omega_y)dt \tag{13}$$

$$\phi \approx \int (\omega_x)dt \tag{14}$$

It is important to note that the absolute angle is not inherently present from the integration of angular rate as from the fundamental integration principle outlined in Eq. (15).

$$\int (\omega)dt = \omega t + C \tag{15}$$

Using the estimated absolute angle from the true gravity vector alleviates this problem as is shown in later observations, but incurs an initial attitude settling time where the attitude estimations must first converge on the angle estimated from the true gravity vector $\vec{a_g}$.

The Laplace domain complementary filter representation is thus shown in Eq. (16).

$$\theta = \left(\frac{1}{1+\tau s}\right)\theta_a + \left(\frac{\tau s}{1+\tau s}\right)\frac{1}{s}\omega \tag{16}$$

Cancelling the s values on the top and bottom of the right hand term simplifies to:

$$\theta = \frac{\theta_a + \tau\omega}{1+\tau s} \tag{17}$$

where θ denotes the filtered estimated angle, θ_a denotes the estimated angle from the accelerometer, ω denotes the estimated angular rate obtained from the gyro, and τ denotes the filter time constant. In order to discretise this continuous equation into one implementable in code, the backward difference substitution $s = \frac{1}{\Delta t}(1-z^{-1})$is used and applied to the denominator of Eq. (17):

$$1 + \tau s = 1 + \tau\left(\frac{1}{\Delta t}(1-z^{-1})\right) \tag{18}$$

$$= \left(1 + \frac{\tau}{\Delta t}\right) - \frac{\tau}{\Delta t}z^{-1} \tag{19}$$

Now inserting the result of Eq. (18) into the original form in Eq. (17) the final discrete realisation can be found. It is important to note that this approach works for both pitch (θ) and roll (ϕ) independently. A constant depending on the time constant and sampling rate is defined as α in Eq. (20) which serves as a unity-sum factor in the complementary filter symbolically representing the degree of trust in the gyroscope data.

$$\alpha = \frac{\frac{\tau}{\Delta t}}{1 + \frac{\tau}{\Delta t}} = \frac{\tau}{\tau + \Delta t} \tag{20}$$

$$\theta_k = \alpha(\theta_{k-1} + \omega_{yk}\Delta t) + (1-\alpha)\theta_{ak} \tag{21}$$

$$\phi_k = \alpha(\phi_{k-1} + \omega_{xk}\Delta t) + (1-\alpha)\phi_{ak} \tag{22}$$

This final discrete time realisation for estimated pitch at time k (θ_k) estimation is shown in Eq. (21). Likewise, the same formula applies to estimating roll at time k (ϕ_k) in Eq. (22).

In order to obtain a reasonable value for the filtering constant α, the gyroscopic drift and accelerometer noise must be characterised. Firstly, the accelerometer is sampled at a rate of 20 Hz and rotated in each axis. The resulting axis angles and frequency spectra obtained by Fast Fourier Transform (FFT) are shown in Fig. 6.

The cursor is placed on the largest magnitude frequency component which occurs at $f = 0.078$ Hz. The corresponding value for τ (in seconds) on the pitch axis is:

$$\tau = \frac{1}{2\pi(0.078)} = 2.04045 \tag{23}$$

Therefore the filter cut-off is $f_c > 0.078$ Hz, and as a result $\tau < 2.04045$ s.

The same procedure is applied to the roll axis, where the cursor is

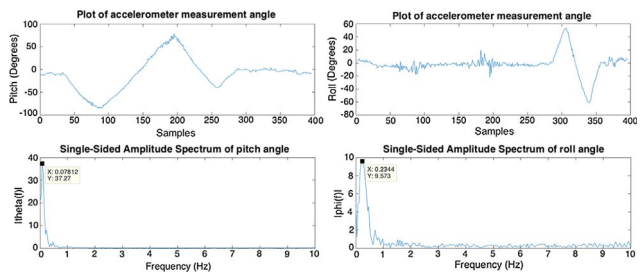


Fig. 6. Plot of measured pitch and roll with frequency spectrums sampled at 20 Hz.

placed on the largest magnitude frequency component which occurs at $f = 0.2344$ Hz. The corresponding value for τ (in seconds) on the roll axis is:

$$\tau = \frac{1}{2\pi(0.2344)} = 0.67898 \tag{24}$$

It follows that the filter cut-off is $f_c > 0.2344$ Hz, and as a result $\tau < 0.67898$ s. Finally an appropriate τ value corresponding to both axes pitch and roll can be determined:

$$\therefore \tau < 0.67898 \tag{25}$$

Thus, if we assume a sampling period of $\Delta t = 0.01$ s for the AHRS, the filter coefficient α can be calculated from Eq. (20) as:

$$\alpha = \frac{\tau}{\tau + \Delta t} \approx \frac{0.678}{0.678 + 0.01} \approx 0.985 \tag{26}$$

This puts the complementary low-pass cutoff frequency for the accelerometer component at just over 0.24 Hz. As a result of the complementary filter structure, this also places the high-pass filter cut-off region for the gyroscope at 0.24 Hz. The gyroscope drift must be analysed in order to carry on with this assumption. If it is found that the primary constituent of gyroscope drift is a constant bias offset then this assumption will hold. The effect of gyroscopic drift is thus measured while the IMU is placed on a flat surface and plotted in Fig. 7.

The effect of gyroscopic drift can be seen in Fig. 7. In this test, sampled gyroscope values were integrated to find the pitch and roll angles for a free-standing IMU. The gyroscope drift effect is apparent in that after only a few seconds the values have diverged. In the frequency

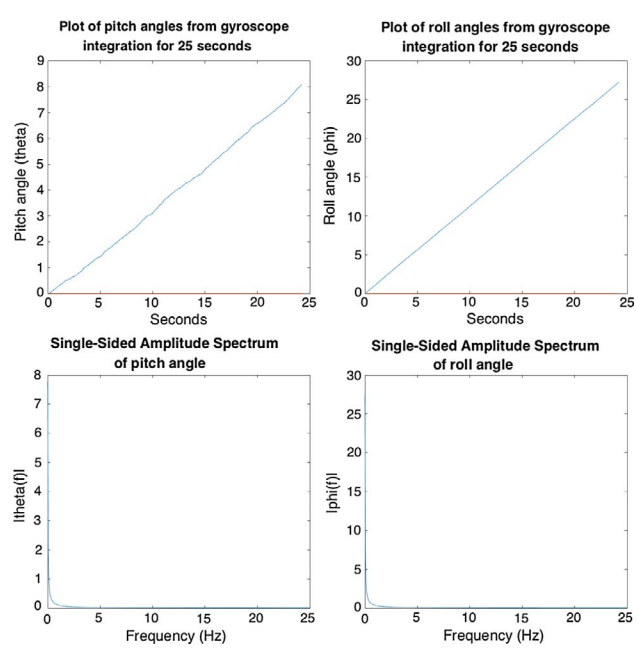


Fig. 7. Plot of measured results of gyro drift on pitch and roll axes over 25 s sampled at 50 Hz.

domain this drift is present in the lower frequencies. More specifically at the 0 Hz region. This implies that the gyroscopic drift can be assumed as a constant bias offset, enforcing the previous assumption of the complementary filter and that a high-pass filter on the gyroscopic component with a cut-off frequency in the region $f_c = 0.24$ Hz should remove the gyroscopic drift effect.

The magnetometer provides a magnetic vector $\overrightarrow{m}$ that points to the Earth's magnetic poles. It is possible to estimate the aircraft magnetic heading (ψ) relative to the poles using the atan2 method described in Eq. (3). Accurate as this may be for applications where the magnetometer is placed on a flat surface, dynamic applications (such as those present on a UAV) with varying bank angles require tilt compensation and additional magnetic interference compensation to be effective.

The method described in [16] utilises knowledge about the pitch (θ) and roll (ϕ) angles to compensate for tilt. The tilt compensated magnetometer values ($\overrightarrow{m_c}$) are given by the rotation matrix:

$$\overrightarrow{m_c} = \begin{bmatrix} \cos(\theta) & \sin(\theta)\sin(\phi) & \sin(\theta)\cos(\phi) \\ 0 & \cos(\phi) & -\sin(\phi) \\ -\sin(\theta) & \cos(\theta)\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \overrightarrow{m} \tag{27}$$

Since the correction in the magnetometer z-axis is not required (because Eq. (3) makes use of only the x and y components of $\overrightarrow{m}$) the tilt compensation equations are simplified to:

$$m_{cx} = m_x\cos(\theta) + m_y\sin(\theta)\sin(\phi) + m_z\sin(\theta)\cos(\phi) \tag{28}$$

$$m_{cy} = m_y\cos(\phi) - m_z\sin(\phi) \tag{29}$$

Magnetometers are also subject to magnetic influence in two ways: hard-iron effects and soft-iron distortion. Additive fields (such as those from a electric motor coil) contribute to hard-iron effects and distortion of the Earth's magnetic field (such as the absorption of magnetic fields from a battery) contribute to soft-iron effects.

Fig. 8 illustrates the effects of hard-iron and soft-iron effects on magnetometer readings. The offset centroid of the samples is due to hard-iron effects. The soft-iron distortion is varied by orientation relative to the sensor and magnetic field. This creates a varied magnitude in each direction, thus contributing to the ellipsoidal distortion of the samples evident in Fig. 8. In order to remove the bias offset as a result of hard-iron effects, a bias error vector due to hard-iron effects is defined $\overrightarrow{B_h}$. By estimating the bias error vector, the hard-iron effects can be removed by:

$$\overrightarrow{m_h} = \overrightarrow{m} - \overrightarrow{B_h} \tag{30}$$

Vectors for keeping track of the maximum and minimum recorded

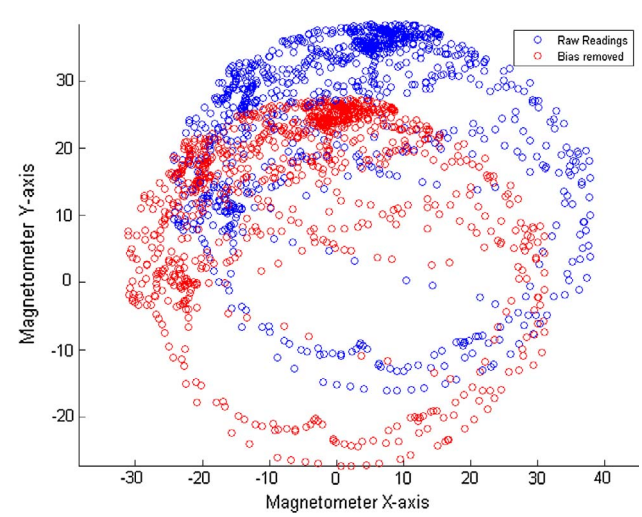


Fig. 8. Scatter plot of magnetometer readings with and without hard-iron compensation applied. Axes made equal to illustrate soft-iron distortion.

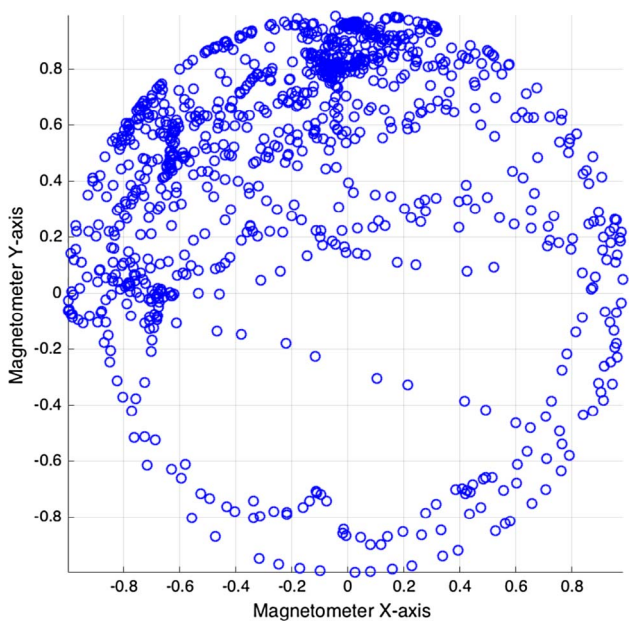


Fig. 9. Scatter plot of magnetometer readings with both hard- and soft-iron compensation applied. Notice the reduction in ellipsoidal distortion.

raw magnetometer readings on all axes (x,y,z) are defined as $\overrightarrow{m}_{max}$ and $\overrightarrow{m}_{min}$ respectively. The bias vector is then:

$$\overrightarrow{B_h} = \frac{1}{2}(\overrightarrow{m}_{max} + \overrightarrow{m}_{min}) \tag{31}$$

Removal of soft-iron effects is achieved by normalising the bias-compensated vector $\overrightarrow{m_h}$ to achieve a vector of magnetometer readings with hard-iron and soft-iron effects removed ($\overrightarrow{m_{hs}}$):

$$\overrightarrow{m_{hs}} = \frac{\overrightarrow{m_h}}{|\overrightarrow{m_h}|} \tag{32}$$

Readings of the magnetometer were recorded by rotating the IMU along all three axes for a total of three revolutions. The hard-iron compensation described in Eq. (30) above was then applied to the data set and the result plotted in Fig. 8. Note that the z-axis is omitted in the plot to better illustrate the offset.

Notice how in Fig. 8, the hard-iron offset is effectively removed by the hard-iron compensation but soft-iron ellipsoidal distortion is still present. Soft-iron compensation as proposed in Eq. (32) was then applied to the same dataset and the result plotted in Fig. 9. Notice how the ellipsoidal distortion from soft-iron effects is removed almost entirely, and the readings are no longer offset by hard-iron distortion.

Finally, the tilt compensation as in Eq. (27) is applied to magnetometer data recorded while rotating the IMU on the pitch and roll axes. Care must be taken to maintain the heading of the IMU while performing the rotations, but some errors are expected due to human error. The result of tilt compensation is shown in Fig. 10. The standard deviation is also displayed in the plot legend, showing a substantial decrease in the error due to tilt from 34.78 degrees to a more acceptable 7.04 degrees.

It is lastly important to note the deviation between magnetic heading (facilitated by the magnetometer) and true heading (used for geographic navigation). The magnetic heading points towards the Earth's magnetic poles, which are slightly offset from the true poles. This deviation is termed the magnetic variation (or declination), which varies between regions and is obtained from the GPS module. Eq. (33) describes the process of removing the easterly magnetic variation (ψ_{ve}) to find the true heading (ψ_T):

$$\psi_T = \psi + \psi_{ve} \tag{33}$$

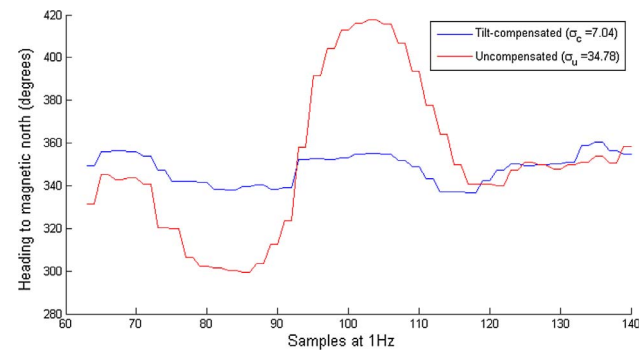


Fig. 10. Plot of heading from magnetometer with and without tilt compensation (IMU kept pointing towards magnetic north).

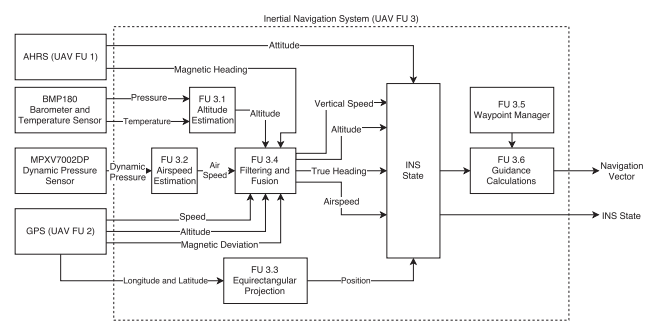


Fig. 11. Functional block diagram of inertial navigation system constituent components.

3.2. Inertial navigation system

The crucial sensor components of the inertial navigation system are primarily the airspeed and altitude estimation. The overview of the inertial navigation system is shown in Fig. 11.

The general algorithmic flow for determining altitude is shown in Fig. 12 below. First, the sensor calibration data is read from the BMP180 device, after-which the uncompensated temperature and pressure can be retrieved from the device. Using the calibration data and measurements, the true pressure and temperature can be calculated and thus the altitude above sea level can be estimated.

The algorithms for pressure and temperature compensation are well described along with pseudo-code in the BMP180 Barometric sensor data sheet [17]. Using these compensated values, the altitude above sea level can be estimated with Eq. (34) using the compensated pressure p (Pa) and the pressure at sea-level p_0 (Pa):

$$a_{asl} = 44{,}330 \times \left(1 - \left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right) \tag{34}$$

The pressure at sea-level (p_0) can be defined by the user at each start-up of the system. However, this approach requires too much user intervention and if entered incorrectly would provide unreliable data. The p_0 value varies over time as the climate of the area changes. It is also true that there are few sources for good atmospheric pressure at sea-level measurements for areas that are not at or around airports. The design decision was thus taken to assume the p_0 value is equal to the standard atmospheric pressure at sea-level. So, p_0 = 101,325 (Pa).

Given this assumption, it can be expected that the estimated altitude

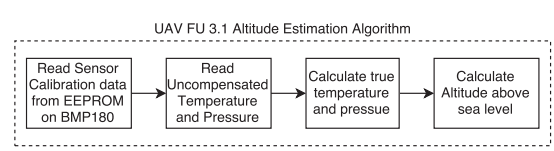


Fig. 12. Altitude estimation algorithm.

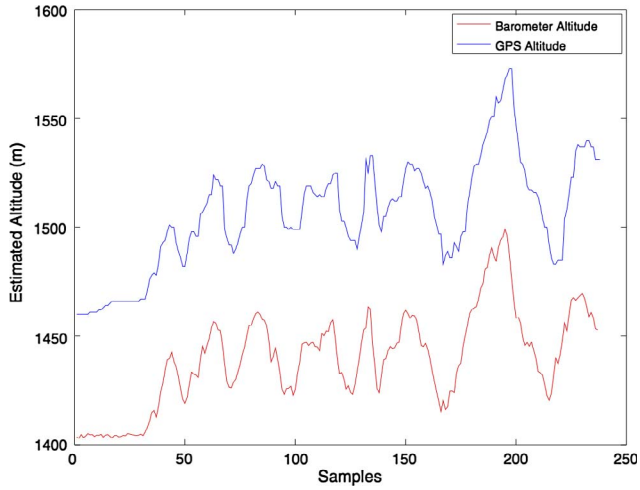


Fig. 13. Plot of GPS altitude vs measured barometer altitude sampled at 1 Hz.

will deviate from the true altitude of the UAV. This deviation can be measured when the UAV is in the start-up phase by comparing GPS altitude with the estimated altitude from Eq. (34). In order to test this assumption, the necessary components were mounted on the UAV and a test flight was conducted with telemetry data. The result of this test flight is shown in Fig. 13 below.

As can be seen in Fig. 13, the measured altitude from the barometer approximately matches the GPS altitude, with the exception of a slight offset due to the assumed p_0 value. The proposed solution is to compensate for this offset by comparing GPS altitude with measured altitude at start up, and subtracting this offset from the estimated altitude in subsequent calculations.

Airspeed estimation uses measured sensor data from the MPXV7002DP dynamic pressure sensor mounted in the forward section of the UAV fuselage to estimate the aircraft airspeed (V_{air}). Air flows into the dynamic and static pressure ports of the aluminium pitot tube to the relevant ports on the MPXV7002DP dynamic pressure sensor via silicon tubing. It is important to note the relation between airspeed (as from a pitot tube) and ground speed (as from the GPS) and why the two are characteristically different.

Ground speed represents the speed of the UAV over ground, whereas airspeed represents the speed of the UAV through air. Airspeed is more relevant in aerodynamics, because it is one of the fundamental aspects that create lift over the wings and alter the flight characteristics of the aircraft. The MPXV7002DP data sheet [18] provides the relationship between measured analog voltage and dynamic pressure. This relationship is shown in Fig. 14.

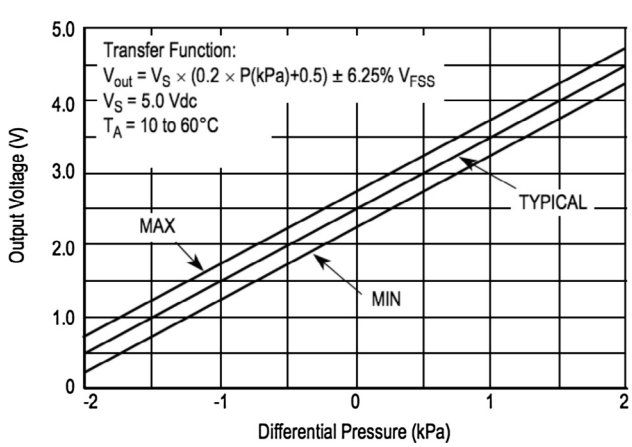


Fig. 14. MPXV7002DP voltage output versus pressure differential [18].

Using the 12-bit ADC on the STM32F415RG, the value can be measured and then converted to a voltage equivalent (V_{dp}) using Eq. (35) below. V_{error} is a product of sensor characteristics as described in the data sheet [18].

$$V_{dp} = 3.3 \times \left(\frac{adcvalue}{4095}\right) - V_{error} \tag{35}$$

Using a supply voltage from the power supply on the electronic speed controller ($V_s = 5$ volts), the equation for estimating dynamic pressure (P in Pascals) from the analog voltage input (V_{dp}) from the MPXV7002DP sensor becomes:

$$P = 5 \times \left(\frac{V_{dp}}{5} - 0.5\right) \times 1000 \tag{36}$$

From the dynamic pressure (P), the airspeed (V_{air} in m/s) can be estimated using the dynamic pressure equation [19] where ρ is standard air density defined as 1.225 kg/m^3:

$$V_{air} = \sqrt{2\left(\frac{P}{\rho}\right)} \tag{37}$$

In order to determine the V_{error} term for Eq. (35) above, a flight test was conducted to record both ground speed from the GPS as well as the raw ADC readings from the airspeed sensor in order to test the accuracy of the design. It was found that $V_{error} \approx 0.25$ V, which falls within the predicted range for V_{error} as outlined in the data sheet [18]. The relevant Eqs. (35)–(37) were applied to the data and subsequently plotted in Fig. 15 below using *Octave*. The result was a successful estimation, however ground speed and airspeed are not always equal due to wind effects. The noticeable zero value noise in Fig. 15 is due to the fact that no filtering was applied. Subsequently these errors were minimised by implementing a simple moving average with five historic airspeed values.

After this altitude estimation, the vertical speed (VS) is determined from the discrete differentiation of estimated airspeed a at time k where Δt is the time between samples:

$$VS_k = \frac{a_k - a_{k-1}}{\Delta t} \tag{38}$$

It is important to note that due to the high accuracy of the BMP180 sensor the readings present are inherently noisy. This noise can have a profound effect on the vertical speed, where the derivative of altitude (vertical speed) is further exaggerated by spontaneous noise values. In

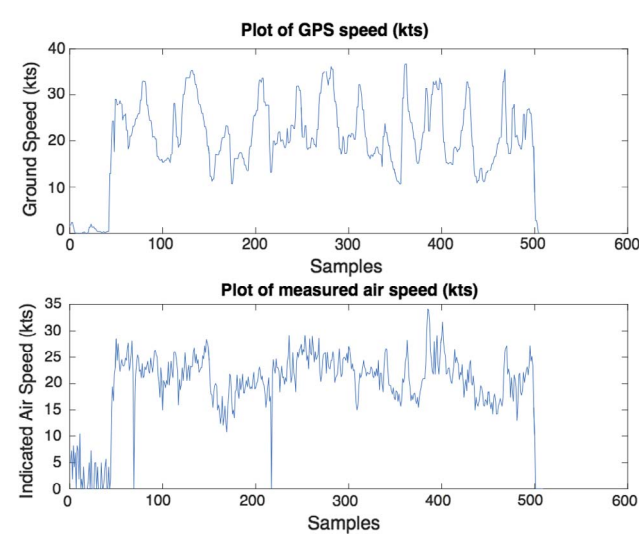


Fig. 15. Plot of free-standing barometer measurements and derived vertical speed values before and after SMA filtering sampled at 10 Hz.

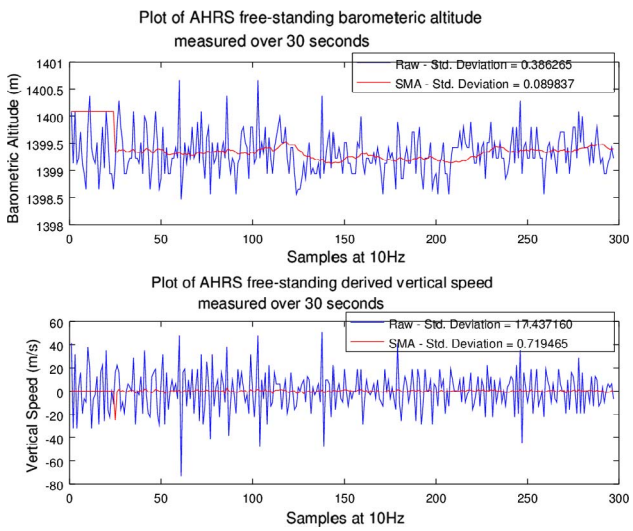


Fig. 16. Plot of ground speed and estimated airspeed sampled at 1 Hz.

order to combat these characteristics, a simple moving average filter was implemented to work on 25 historic sample values ($N = 25$).

In free-standing tests, the BMP180 barometric pressure sensor showed noise in the form of altitude fluctuations as can be seen in Fig. 16. With the proposed SMA filtering, the standard deviation for the derived vertical speed was reduced from an unusable 17.4 m/s for the raw sensor values to a more manageable 0.719 m/s.

Guidance is facilitated by using a line-of-sight guidance approach. By calculating the difference between two position vectors for the UAV's current location and the position of the next waypoint, a set bearing to intercept the waypoint as well as a vertical climb rate can be followed by passing these variables to a flight control system. The position of the UAV is received from an on-board GPS module which returns a geographic coordinate fix. Geographic coordinates are unfavourable for use in navigation due to their non-linear characteristics. It is possible to estimate a small area in a Cartesian representation using geographic latitude λ and longitude φ (not to be confused with roll (ϕ)). Using equirectangular projection which assumes a spherical Earth with a fixed radius (r), geographic coordinates are projected into Cartesian coordinates using:

$$x = r\lambda\cos(\varphi) \tag{39}$$

$$y = r\varphi \tag{40}$$

In order to further reduce the computational complexity, a constant $\cos(\varphi_0)$ is defined. This constant is determined at the home location of the UAV. It is important to keep this constant defined equally between the ground control station and the UAV so as to minimise the distortion on the ground control station map.

3.3. Flight control

The flight control UAV functional software unit is responsible for maintaining the desired flight parameters for orientation as provided by the inertial navigation system. This flight control system is composed of multiple discrete PID controllers working in parallel as is shown in the conceptual overview in Fig. 17.

In order to accurately estimate the flight dynamics, a three-dimensional model was designed for simulation within *Laminar Research X-Plane*. The final design of the model is shown in Fig. 18 below, as well as the implemented autopilot simulator for PID tuning and prototyping. The blade element theory components of lift can be seen on the wing, indicated by green vectors perpendicular to the wing surface.

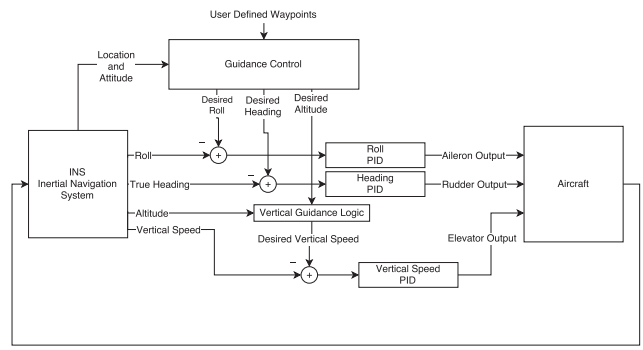


Fig. 17. Conceptual overview of autopilot navigation system.

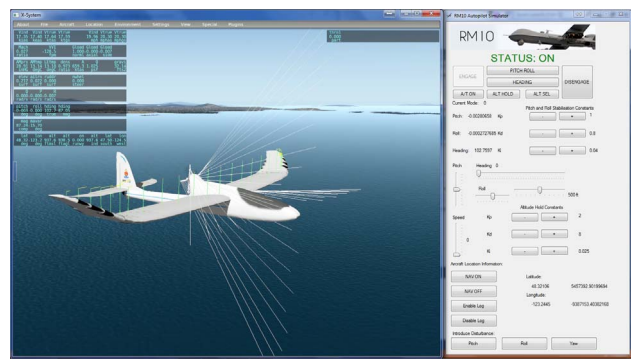


Fig. 18. Screenshot of implemented three-dimensional flight dynamics model of the AXN Floater for X-Plane simulation and implemented autopilot simulator (right).

The Ziegler-Nichols approach was initially used to tune the PID controllers, but this approach proved to produce a rapid response control system not suitable for smooth flight. As a result, the decision was made to tune the PID parameters manually by observing the outputs and oscillations in the simulation setup. For each axis control system the following methodology was applied to tune the controller:

(1) Set all gain values to zero.
(2) Increase proportional gain until a stable oscillation is induced by a disturbance.
(3) Increase the derivative gain until the oscillation is eliminated.
(4) Repeat the above steps (2–3) until a suitable response is achieved that does not induce acceleration forces exceeding 2 g on the airframe (the limit of the LSM303 accelerometer range [20] in high accuracy mode).
(5) Increase integral gain until the offset error is removed.

Using this approach, the parameters for the PID constants were determined and tabulated in Table 1 below to be ready for implementation on the micro-controller.

The discrete PID controller pseudo code is described in the pseudo code in Algorithm 1.

Table 1
Table of obtained PID constants.

Axis Controller	P	D	I
Pitch	1	0.8	0.04
Roll	1	0.8	0.04
Yaw	1	0.8	0.2
Vertical Speed	2	8	0.025
Airspeed	1	14	0.1

Algorithm 1. Discrete PID Controller Algorithm

```
procedure PID curr, set                ▷Apply PID control to curr variable to achieve set variable
    error_k ← set − curr
    if abs(sum) < windup then          ▷windup is maximum sum to prevent integral wind up
        sum ← sum + (error_k × dt)     ▷dt is time in seconds between PID algorithm calls
    else
        sum ← 0
    end if
    derivative ← (error_k − error_{k−1})/dt
    error_{k−1} ← error_k
    drive ← (KP × error_k + KI × sum + KD × derivative)
    if drive > 90 then                 ▷Output should not exceed servo angle limits
        drive ← 90
    else if drive < −90 then
        drive ← −90
    end if
    return drive                       ▷Return output drive for servo
end procedure
```

The described approach (Algorithm 1 above) was used to define the PID control functions for pitch, roll, course and vertical speed – `PID_pitch`, `PID_roll`, `PID_course` and `PID_vs_hold` respectively.

3.4. Fire detection

Using a combination of low-cost air composition sensors, namely the analog MQ-2 (smoke) and MQ-7 (carbon monoxide), a rudimentary fire detection system can be designed. However, due to the variability of these sensors in different humidity and temperature conditions a more elaborate approach must be used. The decision was made to use these sensors in conjunction with the on-board temperature sensor in a neural network classifier to improve robustness.

The effects of temperature and humidity on both the MQ-2 and MQ-7 sensors is illustrated in Figs. 19 and 20 where it is apparent that temperature and humidity both have a substantial effect on the readings. In order to alleviate these effects to create a robust classification system, fire detection is facilitated by using a feed-forward neural network. Due to the long training times associated with neural networks, all training on the data set is pre-performed using an implemented *Octave* training script. In order to train the network, data samples are collected by starting a small contained fire. The samples from the various air composition sensors are recorded for open air as well as smoke by placing the sensors over the chimney of the tin. One thousand samples were recorded for each case – fire and no fire. The training approach used is squared error back propagation with the arc tangent sigmoid function, as proposed and formulated by Bernard Widrow [23] illustrated in Fig. 21. The proposed neural network scheme uses an input layer (3 neurons), hidden layer and output layer (2 neurons) to aid in prediction. Using the *Octave* training script developed from first principles, an optimised set of neural network parameters can be determined.

Using the samples collected, the neural network was trained and the coefficients of each neuron weight recorded for implementation on the micro-controller. A variable approach to the network parameters allowed for various iterations through of the number of hidden neurons in order to obtain the best network parameters. Fig. 22 shows the results found in training the neural network for varying hidden neuron counts. It was found that the optimum number of neurons in the hidden layer is three.

3.5. Motion planning

Motion planning is described as the construction of discrete motions to satisfy a desired motion in three-dimensional space according to

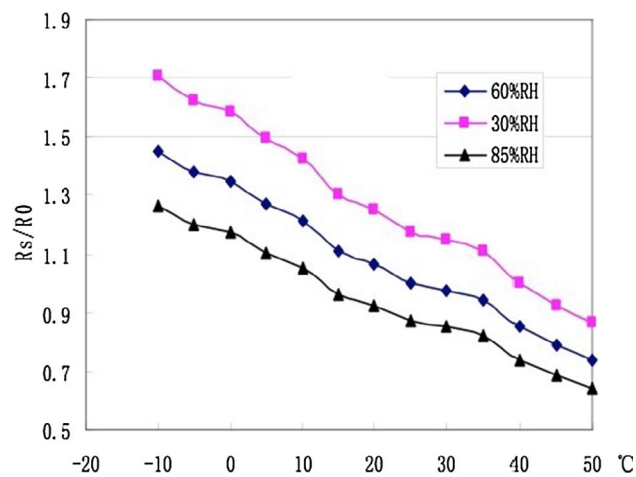


Fig. 19. Typical temperature and humidity effects on MQ-2 sensor resistance. Figure from Hanwei Electronics Co. [21].

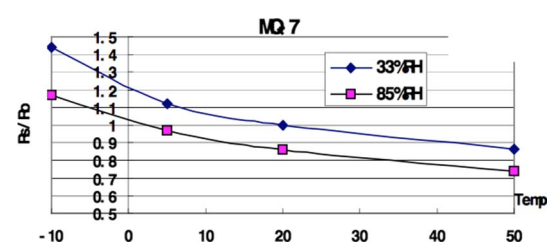


Fig. 20. Typical temperature and humidity effects on MQ-7 sensor resistance. Figure from Hanwei Electronics Co. [22].

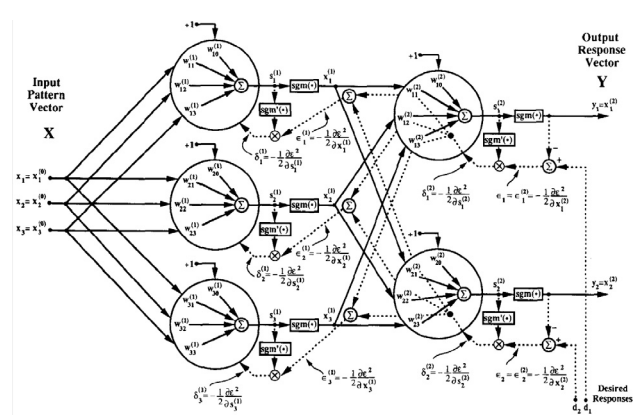


Fig. 21. Illustration of proposed Widrow-based mean square back-propagation neural network. Figure from Widrow [23].

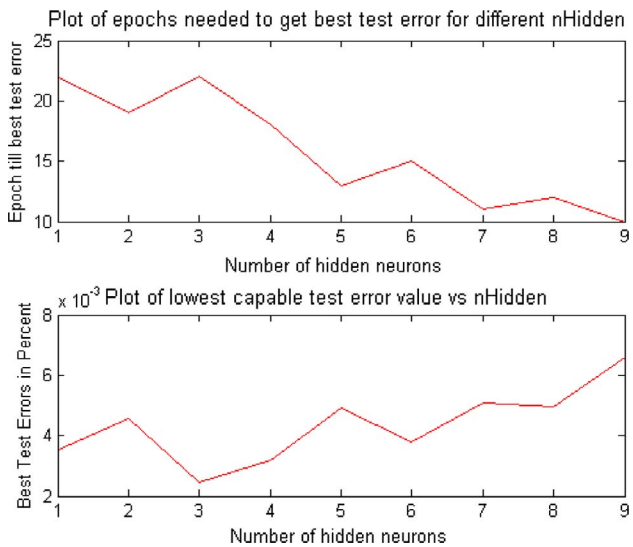


Fig. 22. Plot of number of epochs required for training vs number of hidden neurons and best test error vs number of hidden neurons.

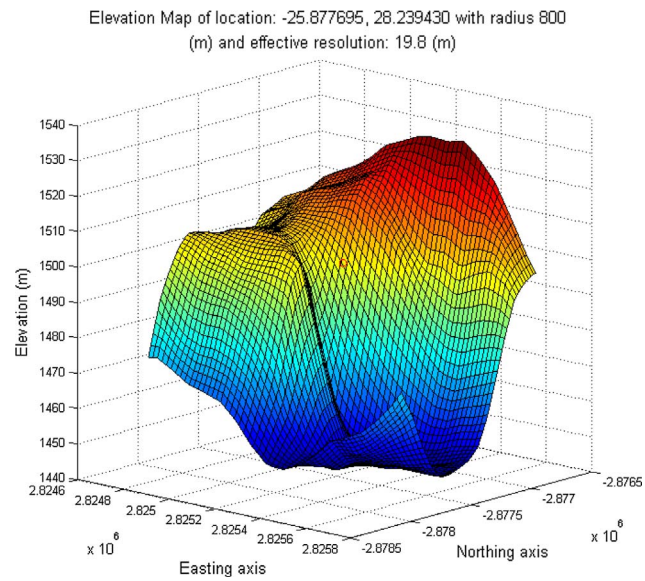


Fig. 24. Typical elevation mesh for an enclosed area showing elevation

certain constraints. Motion planning has become an ever-important aspect of UAV development with the rising interest in autonomous UAV systems. In this respect, artificial intelligence applications for robotic path planning provide a solid starting point for designing a three-dimensional approach to robotic motion planning suited to the UAV application. The two greatest challenges to the application UAV operating autonomously are the ability to navigate mountainous terrain safely and the ability to maximise efficiency by limiting climbs where significant energy loss is possible.

The motion planner has a few main constituent methods. Firstly, the standard convention geographic coordinate system must be linearised through equirectangular projection (FU 2.1 in Fig. 23). After this, the terrain mesh can be constructed using elevation information from a local database (FU 2.2). A safety height is added to ensure the UAV is at a minimum safety height above ground to avoid trees and other obstacles, after which the optimal path is determined by the graph search algorithm (FU 2.3). The optimal path is then re-projected into the geographic coordinate convention (FU 2.4) and output is sent back to the user interface where it can be loaded onto the UAV guidance system.

In order to navigate terrain safely, the approach has been taken to use predetermined three-meter-high accuracy NASA topology information. This information gives the system an idea of the general terrain conditions and when combined with an additional safety-above-ground altitude limit, it provides a safe alternative to radar-based collision avoidance systems. Using predetermined topology data does not incur any additional cost to the UAV, whereas implementation of even a simple LIDAR system can surpass the airframe itself in cost. Despite the accuracy trade-off, the topology data approach suits the application of the UAV which will primarily operate at high altitude. As a result, the high accuracy of the costly LIDAR system is unnecessary. This aspect inherently moves responsibility for complex motion planning to the ground control station computer rather than the UAV micro-controller due to the additional memory needed on the micro-controller to fulfil this aspect. As a result, complex motion planning will be the responsibility of the ground control station software.

In order to perform motion planning between two user-defined waypoints, the general flying area must be defined as a function of the two points, from where the area is then segmented into discretely fixed distance points. The relevant terrain elevation is extracted from the NASA SRTM database and combined with the coordinates to form a three-dimensional point referred to as a node. The collection of nodes that constitute the flying area is referred to as a mesh – an example of which is illustrated in Fig. 24. The mesh contains all the information necessary to formulate a three-dimensional flight plan using graph-search algorithms. The A^* Search is one of the best heuristic driven search algorithms in robotic path planning [24] and as a result it is the chosen graph-search algorithm to be implemented. Formulating a heuristic that takes both distance and change in altitude into consideration can provide an efficient and fast implementation for motion planning. The process described is applied two waypoints at a time for the entire waypoint list – this means that an optimal path is found for each segment of the UAV flight path.

First, in PC FU 2.1 the input user waypoint list (from the graphical user interface) is converted into the equirectangular projection equivalent by converting each individual waypoint into its

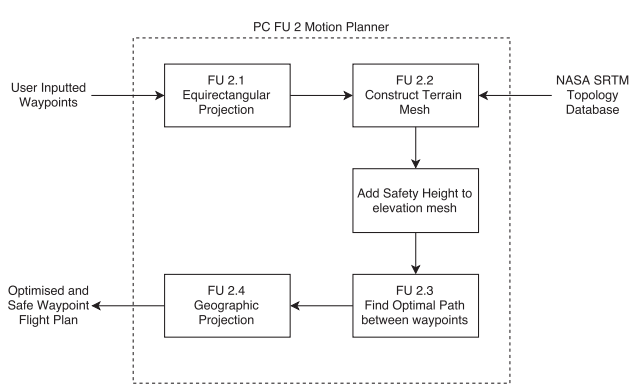


Fig. 23. Conceptual overview of motion planner logic.

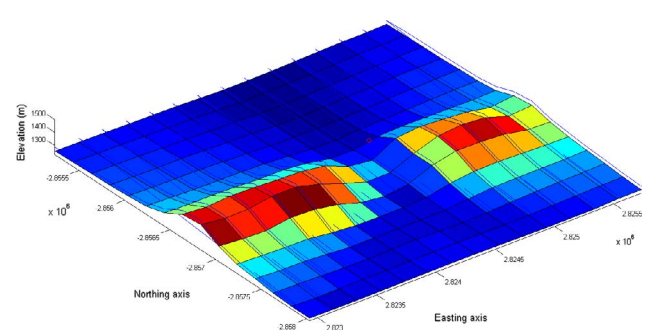


Fig. 25. Plot of generated three-dimensional terrain mesh showing safety height layer for the Wonderboom ridge with accuracy set to 200 m – ie. each block is 200 m by 200 m.

equirectangular equivalent using Eqs. (39) and (40).

Next in FU 2.2, extraction of the terrain elevation at a given point is facilitated by reading the NASA SRTM database, which is composed of multiple files representing separate latitude and longitudes stored in a binary file format. A data reader for the purpose of extracting the terrain elevation had to be developed as seen in Algorithm 2.

Algorithm 2. NASA SRTM Database Data Reader

```
procedure ELEVATION(lat,lng)                                   ▷Get elevation at latitude and longitude
    fileLat ← ceil(abs(lat))
    fileLng ← floor(abs(lon))
    localLat ← round((fileLat−abs(lat)) × 1201)                ▷1201 = No. of database file rows
    loaclLng ← round((abs(lng)−fileLng) × 1201)
    byteLocation ← ((1201−localLat−1) × 1201 × 2) + localLng × 2
    fileBytes ← readAllBytes(fileLat,fileLng)                  ▷Read all bytes from file
    elevation ← (fileBytes[byteLocation] ≪ 8)|(fileBytes[byteLocation + 1])
    return elevation
end procedure
```

Given the equirectangularly projected user waypoint list, the motion planner begins the process of constructing the terrain mesh. First, an origin point at the midpoint between the two waypoints in question is determined. From this origin point four grid limit corners are defined by constructing a square with side length equal to the distance between the two waypoints in question.

Once these two-dimensional grid limits are determined, the grid is sectioned according to a defined minimum accuracy distance. The number of grid segments is dependant on the accuracy desired and the distance between the two waypoints is shown in Eq. (41). The desired accuracy is also the distance between grid points.

$$noGridSegments = \frac{d}{accuracy} \tag{41}$$

The elevation at each point in this two-dimensional grid is then determined by reading the SRTM Database or alternatively, if an Internet connection is available, it can be obtained by requesting from the *Google Maps Elevation API*. At this point, the grid now becomes a three-dimensional terrain mesh of nodes. The elevation values for each node in the terrain mesh are then offset by adding an additional 30 m safety height to avoid trees and other obstacles. These nodes can now be used to plan an effective flight plan by using optimal path detection. Fig. 25 illustrates a terrain mesh constructed for the Wonderboom Ridge in North Pretoria. Note the valley corresponding to the M1 highway. The intersections of lines are the nodes, the parallel running blue lines illustrate the 30 m safety height. During flight, the aircraft should never descend below the safety height over ground.

Once the terrain mesh has been constructed, the A^* search algorithm of PC FU 2.3 can be applied to the graph represented by the terrain mesh nodes. Since the A^* search is a heuristic search algorithm, it is important to define a suitable heuristic. The proposed heuristic is to use the straight-line two dimensional distance between the two points (Eq. (42). This is chosen instead of the three-dimensional distance or any other variations because it will not overestimate the distance between any node and the goal – this is an important factor in the A^* search algorithm which is only optimal when it has an admissible heuristic [25]. Therefore the cost from the current node n at (x_n,y_n) to the goal node at x_g,y_g is $h(n)$ where:

$$h(n) = \sqrt{(x_g - x_n)^2 + (y_g - y_n)^2} \tag{42}$$

The cost from the start node to the current node is expressed differently. The loss of energy due to climbing is experienced mostly due to a change in potential energy (ΔU) [26–28] where $\Delta U = mg\Delta h$, and m is the mass of the UAV, g the acceleration due to gravity and Δh the change in altitude. The distance evaluation function $g(n)$ can take advantage of this fact. In order to reduce the amount of height climbed in the UAV flight path, the distance evaluation function is defined in Eq. (43) below, where n is the current node being evaluated, s is the start node and K_z is a climb penalty factor. The final evaluation function used in the A^* implementation is shown in Eq. (44).

$$g(n) = \sqrt{(x_n - x_s)^2 + (y_n - y_s)^2} + (K_z \times \Delta h) \tag{43}$$

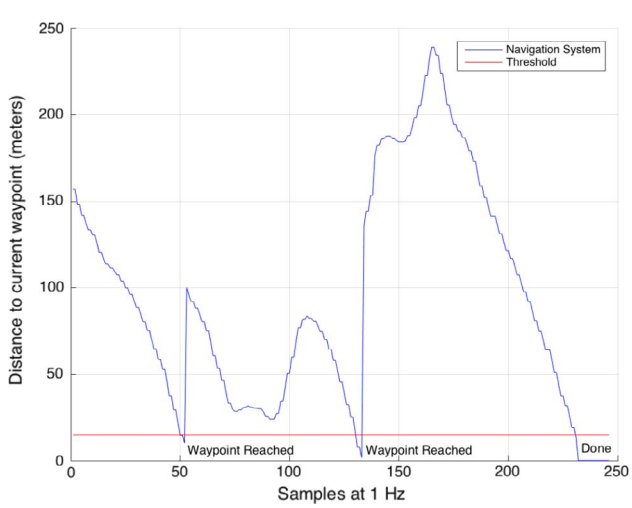


Fig. 26. Plot of calculated distance to current waypoint vs time in seconds. The mission critical minimum interception distance is shown by the red threshold line.

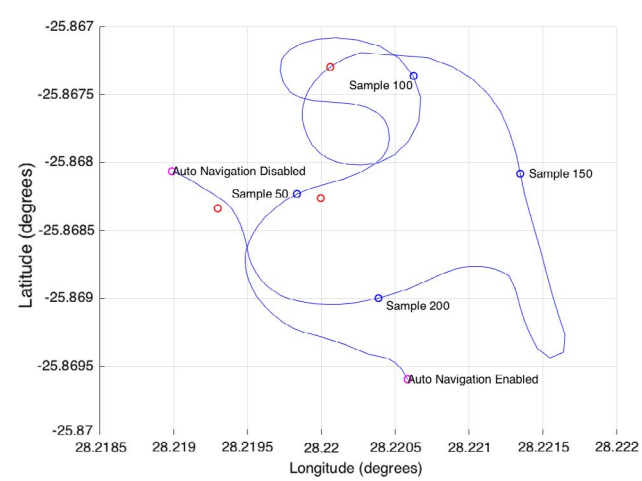


Fig. 27. Flight plot of aircraft navigation. The blue line shows the aircraft's path. The red dots are the waypoints. Blue dots along flight path with associated sample number labels are indicated for interpretation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

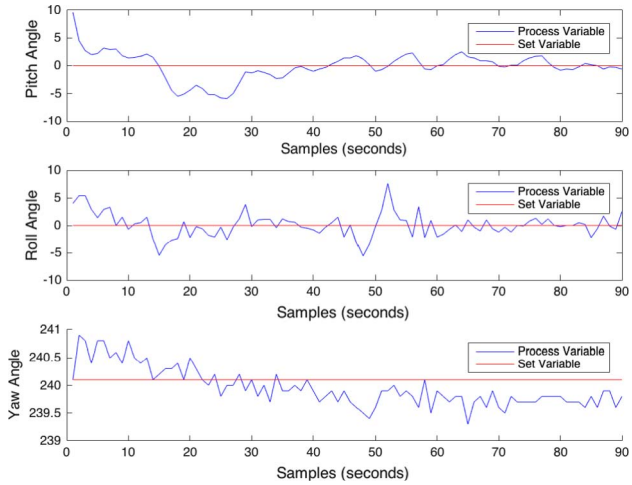


Fig. 28. Plot of PID set and process variables as obtained from test flight telemetry data.

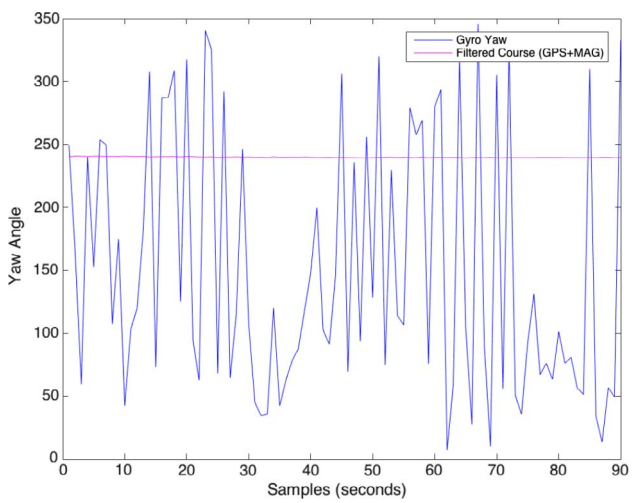


Fig. 29. Plot of estimated yaw from gyroscope integration and the subsequent filtered course obtained from the GPS and magnetometer combination approach.

$$f(n) = g(n) + h(n) \tag{44}$$

In PC FU 2.4, the equirectangular waypoint list is then de-projected back to geographic coordinates using the inverse of Eqs. (39) and (40).

4. Results

4.1. Autonomous navigation

The default telemetry down-link rate of 1 Hz updates as to the INS state and GPS state was enabled. The weather on the day was overcast with an indicated nearby anemometer reading of 14 km/h wind gusts obtained through an Internet weather service. Three waypoints were entered into the ground control station in a triangular shape with less than 150 m between each of them in order to test tight turn situations. The aircraft was manually launched and climbed to a safe height of 1540 m above sea level, after which the autopilot system was engaged on the UAV by using the toggle switch on the radio transmitter. The telemetry results were recorded in the ground control station software as the aircraft navigated and subsequently plotted in Octave.

The average waypoint intercept distance between the UAV and waypoint currently being navigated to is: $\frac{10.1965 + 1.6371 + 13.4142}{3}$ =8.4159 m. The measurement Fig. 26 shows the distance between the UAV and the current waypoint being navigated to. The global minimums indicated by labels show the points where the current waypoint was reached. All

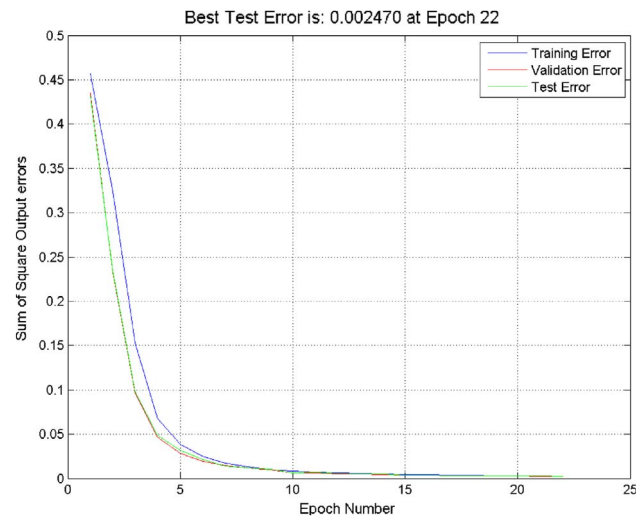


Fig. 30. Plot of learning characteristics for three hidden neuron network at a learning rate $\alpha = 0.02$.

these points are below the mission requirement specification of 15 m accuracy. The apparent oscillations are due to the circling nature of the UAV when a small enough turn radius was not achievable, but ultimately all waypoints were reached. Fig. 27 shows the flight plot of the UAV for the qualification test. The UAV reached the waypoints but performed a loop around the second waypoint after sample 50 in the flight path. Shortly after sample 125 a strong wind gust above mission specifications blew the UAV off course to a distance where it was barely visible. Manual control was assumed and the UAV was flown back into view and the automatic control mode re-enabled. Flying beyond line-of-sight range is illegal in South Africa. The UAV successfully navigated all desired waypoints.

4.2. Flight stabilisation

The setup process from the autonomous waypoint navigation test was repeated but with no waypoints entered. This put the UAV in flight stabilisation mode with course hold and enabled the measurement of the PID control system performance. The flight stabilisation PID test was conducted on a calm day with 7.4 km/h wind speeds. The measurements for flight stabilisation are plotted below in Figs. 29 and 28 below.

It can be seen in Fig. 29, that the approach used in combining the GPS and magnetometer data to provide heading estimation was significantly less noisy than the gyroscope integration method. Fig. 28 shows the PID controller plots for a real flight stabilisation test measured over 90 s. The pitch angle of the UAV is at 9 degrees when the flight stabilisation is engaged, where it then overshoots by 5 degrees and returns to maintain a value around the zero degree set point. The same happens for the roll angle but it settles at around 20 s. Around the 50 s mark a discrepancy can be seen. The yaw angle stays relatively close to the set course but deviates with a mean angle error of 0.1422

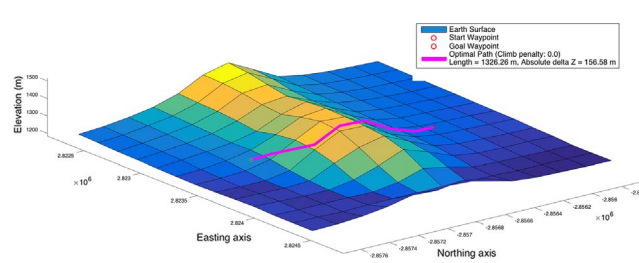


Fig. 31. Wonderboom ridge gorge path Plan with climb penalty $K_z = 0$ and effective resolution < 200 m.

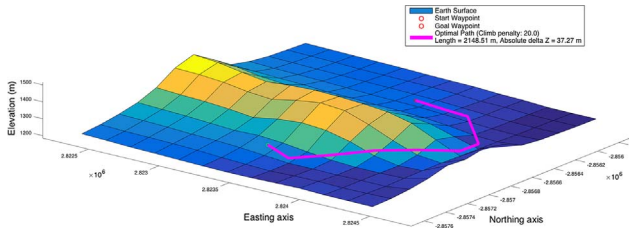


Fig. 32. Wonderboom ridge gorge path plan with climb penalty $K_z = 20$ and effective resolution < 200 m.

Fig. 33. Final unmanned aerial system solution with visible telemetry radio and GPS receiver. Further avionics and microcontroller are mounted internally.

which is slightly higher than the other two axes. A constant offset in the yaw angle process variable can be seen from beyond the 40 s mark. The mean pitch angle error is −0.0978, mean roll angle error is 0.0900 and the mean yaw angle error is 0.1422.

The scheduling approach used to manage parallel running processes allowed the UAV to achieve a flight control stabilisation rate exceeding the specification by 517%. The UAV successfully stabilised itself using on-board PID controller algorithms that were manually tuned using the developed *X-Plane* simulator interface application. The offset error apparent in the yaw angle process variable is due to the fact that integral control was disabled on the yaw axis for test flight purposes. This was because the yaw rate control is much slower than the other axes and the integral term would accumulate, thereby rapidly causing instability. The inconsistency errors in the pitch and roll angle process variables are assumed to be disturbances caused by wind.

4.3. Fire detection

A total of one thousand samples were recorded, half of which were in open air and the other half above the smoke. As was previously determined, the optimal number of hidden neurons is three. The network was trained with a 600 sample training set, 150 sample validation set and 150 sample test set. The results of the implemented training script and learning characteristics are plotted in Fig. 30 below.

The best test error is 0.002470, which correlates with a detection rate of 99%. The plot in Fig. 30 shows the learning characteristics for a feed-forward neural network classification system for detecting fires using back-propagation learning at a learning rate $\alpha = 0.02$. After a single epoch (full back-propagation of error through all training data), the best test error is just below 0.45. After five epochs the test error had reduced significantly. It continued going down through subsequent epochs asymptotically until it reached a minimum test error of 0.002470. After this minimum, the over-fitting condition was triggered, which ended the training and the subsequent best test error was thus achieved at epoch 22.

The neural network implementation managed to achieve a classification accuracy of 99%, which far exceeded the mission-critical specification accuracy of 70%. The value for the learning rate α was deemed appropriate, based on the fact that it did not cause any rapid divergence in the training error plots. Training was relatively short, where the full network was trained to the highest measured accuracy in only 22 epochs. The resulting neural network structure consisted of 3 input neurons, 3 hidden neurons and 2 output neurons – a structure which is small enough to be easily implemented into code with minimal computation time.

4.4. Motion planning

The Wonderboom Ridge M1 highway gorge was chosen as the test scenario area from where to test the motion planning algorithms. An example scenario was created where a hypothetical UAV operator would place a waypoint beyond the ridge. If effective, the UAV path planner would find the shortest path to reach the goal waypoint. In addition to this, the K_z climb penalty factor was varied to show the different approaches the motion planner artificial intelligence would take based on how much energy it is allowed to lose in climbing. The resultant path plans are shown by the three-dimensional magenta path line and terrain mesh in Figs. 31 and 32 below.

Increasing the climb penalty factor K_z from 0 to 20 increased the total path distance by $\frac{(2148.51 - 1326.26)}{1326.26} \approx 62\%$, but decreased the loss in energy from climb by $\frac{(156.58 - 37.27)}{156.58} \approx 76.2\%$. The path plot in Fig. 31 shows a path plan that simply takes the UAV straight over the ridge for the shortest path. The total change in altitude is 156.58 m. The second path plot in Fig. 32 shows a path plot where the artificial intelligence has decided to circumnavigate the mountain ridge in favour of a flatter flight path with less change in altitude by navigating through the gorge. The path planner prevents the UAV from colliding with the terrain by implementing a minimum above-ground height that is apparent in the plots.

It is apparent that the motion planner algorithms that were implemented and proposed work the way they were designed to. The implemented ground control station path planning solution will thus be a valuable tool in increasing the usability and safety of the UAV operations in mountainous areas as well as in situations where the UAV operator has no true idea of the area where the UAV needs to operate (see Fig. 33).

5. Conclusion

The complete system capable of both autonomous navigation and remote fire detection was developed from first principles and it achieved results which fell in line with the desired outcomes. Operational sub-systems such as fire detection and autonomous waypoint navigation surpassed the system specifications with producing optimal results. The final implementation of all sub-systems achieved robust attitude estimation, fully autonomous waypoint navigation, stabilised flight, exceptional fire detection rates and a complete motion planning solution that successfully aided in flight planning over mountainous terrain. The attitude reference and heading system is a strong point of the system since, despite its simplicity, it still maintains the ability to operate under the harsh vibrations and forces experienced in a small unmanned aircraft. However, the use of a more advanced attitude estimation approach could significantly improve the applicability of the system and push the envelope of operational applications further into harsher environments. Flight control managed to stabilise the system adequately in this application too.

6. Future work

In the scope of future work for this implementation, several aspects could be improved upon. Primarily, a Kalman State Estimator filter would have been used in place of the complementary filter. Although the complementary filter approach provides a relatively robust and

accurate attitude estimation, it is not without its faults – primarily the gimbal-lock issue. Although this issue can be circumvented using a quaternion attitude representation, given a wide variety of sensor information and adequate state equations a Kalman state estimator generally provides a better solution.

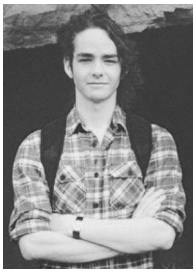
Motion planning could be refined by adapting the proposed heuristic to take into account the UAV flight dynamics. Using a more appropriate model for the effects of drag on the wing tips at different bank angles, a heuristic can be developed to improve the efficiency of path planning turns as opposed to simply optimising the distance and energy loss due to climbs as implemented in this project. By further analysing the efficiencies of flat altitude vs climbs, a more accurate estimation of the improved efficiency can be obtained. The motion planner could further be developed to take air traffic into effect, making safe operation of the aircraft in populated regions more feasible.

Acknowledgment

The research project was supported by the National Research Foundation (NRF), South Africa (Grant Nos.: IFR160118156967 and RDYR160404161474), T-Systems and Intervate (a T-Systems Company), South Africa. The authors would like to thank Prof. Petros A. Ioannou for his advice in improving quality of this research work.

References

[1] SEMI, Sharply falling mems prices spur rising demand, 2010. Available at: <http://semi.org/en/sharply-falling-mems-prices-spur-rising-demand> (accessed 9 March, 2016).
[2] J. Goodyer, Drone rangers, Eng. Technol. 8 (5) (2013) 60–61.
[3] F.P.A. of Southern Africa, Fpasa: Fire statistics 2013, 2013. Available at: <http://www.fpasa.co.za/images/FireStats/Fire_Stats_2013.pdf> (accessed 10 March, 2016).
[4] R. Yanushevsky, Guidance of Unmanned Aerial Vehicles, CRC Press, 2011.
[5] R. Lozano, Unmanned Aerial Vehicles: Embedded Control, John Wiley & Sons, 2013.
[6] A. Wahab, R. Mamat, S.S. Shamsudin, The development of autopilot system for an autonomous uav helicopter model, in: 1st Regional Conference on Vehicle Engineering & Technology, 2006.
[7] M. Euston, P. Coote, R. Mahony, J. Kim, T. Hamel, A complementary filter for attitude estimation of a fixed-wing uav, 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2008, pp. 340–345.
[8] D. Drones, Kalman filter vs complementary filter, 2012. Available at: <http://diydrones.com/forum/topics/kalman-filter-vs-dcm> (accessed 10 July, 2016).
[9] W. Premerlani, P. Bizard, Direction cosine matrix imu: Theory, DIY DRONE, USA, pp. 13–15, 2009.
[10] T. Celik, H. Demirel, H. Ozkaramanli, M. Uyguroglu, Fire detection in video sequences using statistical color model, 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, vol. 2, IEEE, 2006pp. II–II.
[11] V. Vipin, Image processing based forest fire detection, Int. J. Emerg. Technol. Adv. Eng. 2 (2) (2012) 87–95.
[12] H. Mueller, A. Fischer, A robust fire detection algorithm for temperature and optical smoke density using fuzzy logic, in: Institute of Electrical and Electronics Engineers 29th Annual 1995 International Carnahan Conference on Security Technology, 1995. Proceedings. IEEE, 1995, pp. 197–204.
[13] B.J. Meacham, The use of artificial intelligence techniques for signal discrimination in fire detection systems, J. Fire Protect. Eng. 6 (3) (1994) 25–136.
[14] N.A. Chaturvedi, A.K. Sanyal, N.H. McClamroch, Rigid-body attitude control, IEEE Control Syst. 31 (3) (2011) 30–51.
[15] T.G. Gainer, S. Hoffman, Summary of transformation equations and equations of motion used in free flight and wind tunnel data reduction and analysis, 1972.
[16] T. Ozyagcilar, Implementing a tilt-compensated ecompass using accelerometer and magnetometer sensors, Freescale semiconductor, AN 4248, 2012.
[17] BMP180 Data sheet, Infineon Technologies AG, 10 2013, rev. 2.5.
[18] MPXV7002 Integrated Silicon Pressure Sensor On-Chip Signal Conditioned, Temperature Compensated and Calibrated, Freescale Semiconductor, 2015.
[19] L. Clancy, Aerodynamics, chapter 3, Topics on pressure fields, 1975, pp. 22–59.
[20] LSM303DLHC Data Sheet Ultra-compact high-performance eCompass module: 3D accelerometer and 3D magnetometer, ST Microelectronics, 2013.
[21] MQ-2 Semiconductor Sensor for Combustible Gas, HANWEI ELECTRONICS CO. LTD.
[22] Technical Data on MQ-7 Gas Sensor, HANWEI ELECTRONICS CO. LTD.
[23] B. Widrow, M.A. Lehr, 30 years of adaptive neural networks: perceptron, madaline, and backpropagation, Proc. IEEE 78 (9) (1990) 1415–1442.
[24] W. Zeng, R. Church, Finding shortest paths on real road networks: the case for a*, Int. J. Geogr. Inf. Sci. 23 (4) (2009) 531–543.
[25] R. Dechter, J. Pearl, Generalized best-first search strategies and the optimality of a, J. ACM (JACM) 32 (3) (1985) 05–536.
[26] E.S. Rutowski, Energy approach to the general aircraft performance problem, J. Aeronaut. Sci. (2012).
[27] Zhe Tian, Fushun Liu, Zhixiong Li, Yingchun Xie, The development of key technologies in applications of vessels connected to Internet, Symmetry 9 (10) (2017) paper ID. 211.
[28] Johan Wannenburg, M. Reza, Physical activity recognition from smartphone accelerometer data for user context awareness sensing, IEEE Trans. Syst. Man Cybernet.: Syst. 47 (12) (2017) 3142–3149.



Gavin Jaime Fouche completed his B.Eng. in Computer Engineering at the University of Pretoria in 2016. His research interests include aviation, intelligent transportation systems and machine learning.

Reza Malekian is an Associate Professor and Research Group Head in Advanced Sensor Networks at the Department of Electrical, Electronic, and Computer Engineering, University of Pretoria, Pretoria, South Africa. His current research interests include advanced sensor networks, Internet of Things and mobile communications. Dr. Malekian is also a Chartered Engineer registered in Engineering Council of UK, an IEEE Senior member, an ACM Senior member, and a Fellow of the British Computer Society. He is also an associate editor for the IEEE Internet of Things Journal and an editor for IET Networks.