9

Flight Dynamics Modelling and Experimental Validation for Unmanned Aerial Vehicles

X.Q. Chen[1], Q. Ou[1], D. R. Wong[1], Y. J. Li[1], M. Sinclair[1], A. Marburg[2]
[1]*University of Canterbury, New Zealand*
[2]*Geospatial Research Centre (NZ) Ltd, New Zealand*

1. Introduction

Unmanned Aerial Vehicles (UAVs) are a viable alternative to manned aircraft and satellites for a variety of applications, including environmental monitoring, agriculture, and surveying. They promise greater precision and much lower operating costs than traditional methods. Critical to the success of UAV systems is the auto-pilot system which keeps the vehicle in the air and in control in the absence of a human pilot. The development of auto-pilot systems for UAVs is an area undergoing intense research. The ability to test autopilot systems in a virtual (software) environment using a software flight dynamics model for UAVs is significant for development.

A reliable UAV simulation process which can be adapted for different aircrafts would provide a platform for developing autopilot systems with reduced dependence on expensive field trials. In many cases, testing newly developed autopilot systems in a virtual environment is the only way to guarantee absolute safety. Additionally the model would allow better repeatability in testing, with controlled flying environments.

Numerical modelling of flight dynamics has a long history in the aerospace industry, and is used in the development of all modern aircraft and satellites. A flight dynamics model is a mathematical representation of the steady state performance and dynamic response that is expected of the proposed vehicle, in this case a UAV (dcb.larc.nasa.gov/Introduction/models.html). The uses of flight dynamics models are diverse. Commercial, military, government organisations and academic sectors employ flight models to achieve their specific tasks (Chavez et al. 2001). Example applications include control algorithms testing, stability and flight characteristics evaluation of preliminary designs, onboard embedded autopilot systems, and onboard Inertial Navigation Systems (INS).

In the development of UAVs and auto-pilot systems, a flight dynamics model for flight simulations allows rapid and safe testing on a computer. However, a software model developed from first principles has unknown accuracy. For such a model to be of real use, its development process is necessary to include implementation, verification and validation. The approach for each stage of the development process is presented in this chapter.

2. Brief Review of Flight Model Implementation Methods

The fundamental goal of flight dynamics modelling is to represent the flight motion numerically for a given input, as close to the flight motion in the real world as the application requires. To accommodate a wide range of applications, various implementations of flight models in terms of assumptions and algorithms therefore exist. Nevertheless, all flight dynamics models are based on the mathematics model derived from Newtonian Physics. From Newton's second law, an aircraft's motion in its six degrees of freedom (DOF) can be described by a system of non-linear first order differential equations. These equations of motion served as the fundamentals for almost all flight dynamics models. With today's computing power, the processing time of solving these equations is trivial comparing to other signal processing algorithms (e.g. Kalman filter) that might be implemented as part of the flight model. A number of popular numerical techniques used to solve these non-linear systems range from highly efficient to precise but computational intensive: Euler, Heun, Bogacki-shampine, Runge-kutta and Dormand-Prince. As flight modelling becomes ever more sophisticated and more application specific, they are implemented to a vast number of different flavours.

In flight modelling, non-linear model is often used in personal computers where resources are abundant and speed is not a concern. It has been recognized that the significant improvements of dynamic performance of current and new generation of advanced airplanes is possible if flight systems design integrates nonlinear analysis, control, and identification (Lyshevski, 1997). There were several UAV flight control system design projects applied non-linear flight models to simulate the dynamic behaviour of their vehicles Buschmann et al., 2004; Guglieri et al., 2006). On the other hand, the non-linear mathematical model can be linearized to simplify the computation (Ye et al., 2006). In many applications, where the linearized model is considered 'close enough', the required computing power is significantly reduced. For instance, linearized model is widely used in embedded system of micro UAVs, with a wing span of less than 50 cm, to compromise the limited processing power of the onboard computer (Jackowski et al., 2004).

A flight dynamics model may also employ one of the two common orientation representations – Euler angle and quaternion. The most common way to represent the attitude of an aircraft is a set of three Euler angles. These are popular because they are easy to understand and easy to use. However, the main disadvantages of Euler angles are: (1) that certain important functions of Euler angles have singularities, and (2) that they are less accurate than unit quaternions when used to integrate incremental changes in attitude over time (Diebel, 2006). Nowadays, quaternion has been increasingly adapted in flight models because it is better able to avoid singularities and high data rates associated in Euler angle representation (Cooke et al., 1994).

For projects that require a generic flight model, there are some options available. Representative existing models available to the public domain are X-plane (Laminar Research), JSBSim (Berndt J.) and Flight Simulator X (Microsoft). Among these, only JSBSim is open source software, whereas the other two simulation programs are like a 'black box' which does not allow users to access the internal of the models. JSBSim is a 6-DOF non-linear flight model that adopts quaternion angle representation. The open source feature of JSBsim has gained a lot of attention from researchers because of the significant cost saving. A full 6-DOF simulator for flight simulation and pilot training was constructed at the University of Naples using JSBSim as its physics engine (Marco, 2006).

In most projects where time and budget permit, researchers favour to build their own flight models in Matlab and Simulink, or using other generic computer languages like C++ or C for embedded systems (Rasmussen & Chandler, 2002; Berndt, 2004; Buschmann et al., 2004; Jordan et al. 2006) . Apparently building a model from scratch is the best way to have full control over the model, and hence can effectively customize the model to meet their specific applications. Implementing a flight model in Matlab is generally the first step towards development. Matlab provides comprehensive modelling tools that allow concepts to be quickly validated and fine-tuned for improvements. Coding the flight model in C++ or C is normally the final stage when the accuracy of the model is validated and the performance has become a primary concern. The focus of flight model implementation here is on a rapid development approach by integrating Datcom, FlightGear with Matlab Simulink.

3. Integrated Flight Dynamics Modelling Methodology

3.1 Modelling Approach

The first stage of the work was focused on the development of a generic flight dynamics model (FDM). The role of the FDM in the whole simulation is a physics engine that process parameters from all input information. By manipulating input variables mathematically, an FDM predicts the future states of an aircraft. The generic FDM was developed in Matlab Simulink, using the Aerospace Blockset.

Aerodynamic coefficients (AC) characterize the response of the proposed vehicle based on its geometry. With a generic FDM implementation in mind, the ACs are not provided by the FDM and hence need to be determined by other means. As long as the aerodynamic coefficients are available, the FDM models the motion of any vehicle configuration, from a ball (potentially for debugging), to a transonic fighter. The FDM, like any other dynamics model, is a data driven program. Hence the accuracy of its outputs depends on the quality of the input information supplied. The FDM takes initial conditions of the vehicle, and other inputs including aircraft properties (e.g. inertia and gravity), aerodynamic coefficients, control inputs and relative wind conditions. It then outputs the vehicle dynamic responses. Fig. 1. illustrates the internal data flow of the FDM.

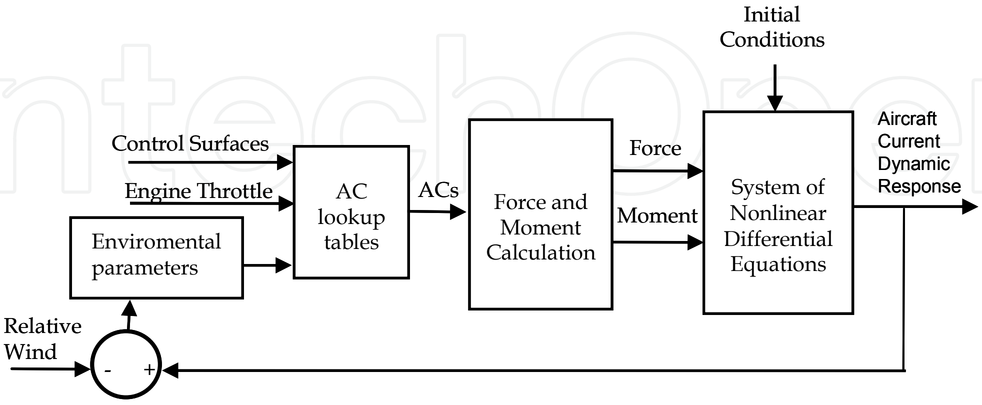


Fig. 1. Flight dynamics model schematic

To establish a model for a particular UAV, the determination of all UAV's aerodynamic coefficients is not a trivial task. Two methods to determine the coefficients are: i) experimental approach, and ii) mathematical approach. The former is generally very accurate, but time consuming and expensive. On the other hand, the mathematical approach is much faster, with little cost and very repeatable by the aid of computers. However, the inaccuracy in the mathematical approach is inevitably caused by the complex aerodynamics involved in aircraft modelling, and the uncertainty in the real environment conditions. The best compromise is to use a combination of these two methods, with the static ACs determined in a wind tunnel and the derivative ACs determined by software.

This work used the software Datcom (Galbraith, 2004) to calculate aerodynamic coefficients from first principles. By writing an input file containing all essential geometries of an aircraft, Datcom produces an output file with aerodynamic coefficients. The interface to Matlab is achieved by a Matlab script file that loads all the essential aerodynamic coefficients from the Datcom output file to the Matlab Workspace. The coefficients in the six degrees of freedom are drag, lift, side, pitching moment, rolling moment, and yawing moment coefficient. By interfacing Datcom with the FDM in the frontend, an aircraft model for any fix wing UAVs can be rapidly developed without wind tunnel testing. This feature significantly increases the repeatability of flight simulation and is found very useful for UAV preliminary designs, where only a rough estimate of the vehicle's stability is required.

Data visualization is another aspect considered while building the flight dynamics model. This has led to interfacing the FDM with an open source flight simulation program FlightGear (Barr, 2006), which can produce a 3D graphic animation in real time. The animation facility allows the UAV to be viewed from any angles, and provides absolute visual information on the UAV attitude and stability. With the FlightGear added in the back-end, the complete simulation environment with the FDM as the physics engine is shown in Fig. 2. This integrated flight dynamics model was developed in the earlier work (Ou, 2008).

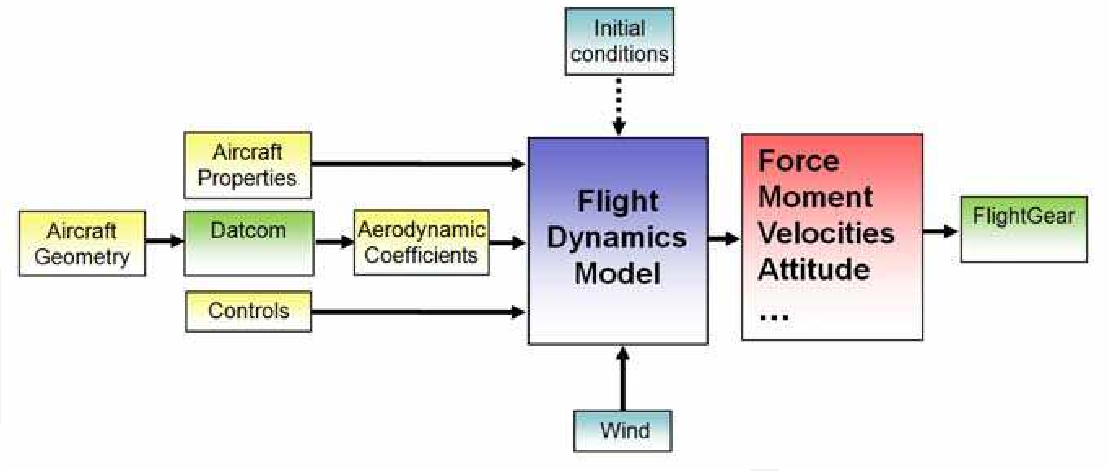


Fig. 2. Block diagram of integrated flight dynamics modelling

3.2 Mathematical Model

Depending on the tool used, the knowledge of underlying mathematics is not always necessary for constructing a flight model. For example, the presented approach in the previous section was to build the model using Aerospace Blockset in Matlab Simulink, where nearly all the functional blocks for rapid aircraft model development are provided. However, in applications that require onboard processing, the understanding of the mathematical model becomes prerequisite. With a mathematical model, the FDM can be implemented in any languages (VHDL, assembly or C language) for embedded systems. Furthermore, transfer functions can be derived from the mathematical model to optimize auto-pilot control strategies.

Essentially, a dynamics flight model can be represented as a system of first order non-linear Differential Equations (DEs). The simulation results are given by solving the system for all the state variables with respect to time. Traditionally, the system of DEs is established based on body coordinates. Response in other coordinate system of interest can be obtained by transformation matrices (e.g. Direction Cosine Matrix). A 6-DOF flight model consists of six fundamental state variables: *u v w* (body velocities), and *p q r* (body angular velocities).

Aircraft properties include aircraft's mass, moment of inertia and centre of gravity. For UAV modelling, it can be assumed the aircraft has constant mass over a flight. The moment of inertia along three axes is a 3×3 matrix denoted:

$$I = \begin{pmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{pmatrix}$$

For traditional aircraft, symmetry and uniformly distributed mass about the x-z plane can be assumed. As a result, the products of inertia $I_{xy} = I_{yx} = I_{zy} = I_{yz} = 0$. Since I_{xz} and I_{zx} are generally very much smaller than I_x, I_y and I_z, a further simplification can be made by neglecting them, so $I_{xz} = I_{zx} = 0$. This assumption is a very satisfactory approximation for UAV models. The accelerations in x, y and z-axis can be expressed in the following equations (Cook, 2007):

$$\dot{u} = rv - qw + \sum F_x / m \tag{1}$$

$$\dot{v} = pw - ru + \sum F_y / m \tag{2}$$

$$\dot{w} = qu - pv + \sum F_z / m \tag{3}$$

Where *r, p, q* are angular velocities about body coordinates x-, y-, and z-axis respectively. F_x, F_y, F_z are force in x-, y-, and z-axis respectively. *m* is the mass of the vehicle. The angular accelerations about x-, y- and z-axis can be expressed as (Cook 2007):

$$\dot{p} = \left[\sum M_x + (I_y - I_z)qr\right] / I_x \tag{4}$$

$$\dot{q} = \left[\sum M_y + (I_x - I_z)pr\right] / I_y \tag{5}$$

$$\dot{r} = \left[\sum M_z + (I_x - I_y)pq\right] / I_z \tag{6}$$

Transformation from body velocities to earth-fixed reference frame velocities is achieved by multiplying the inverse of Direction Cosine Matrix (DCM) (The MathWorks, Aerospace Blockset 3 User Guide).

$$\begin{pmatrix} \dot{X}_e \\ \dot{Y}_e \\ \dot{Z}_e \end{pmatrix} = DCM^{-1} \begin{pmatrix} u \\ v \\ w \end{pmatrix}$$

Where X_e, Y_e, Z_e are x-, y-, z-coordinates in the earth-fixed reference frame, and

$$DCM^{-1} = \begin{pmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix}$$

Where ϕ, θ, ψ are rolling, pitching and yawing angle in earth-fixed reference frame respectively.

Hence, the body velocities to earth-fixed reference frame velocities can be obtained:

$$\dot{X}_e = u\cos\psi\cos\theta + v(\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi) + w(\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi) \quad (7)$$

$$\dot{Y}_e = u\sin\psi\cos\theta + v(\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi) + w(\sin\psi\sin\theta\cos\phi - \cos\phi\sin\phi) \quad (8)$$

$$\dot{Z}_e = -u\sin\theta + v\cos\theta\sin\phi + w\cos\theta\cos\phi \quad (9)$$

The rates of rolling, pitching and jawing angles are given as follows (The MathWorks, Aerospace Blockset 3 User Guide):

$$\dot{\phi} = p + (q\sin\phi + r\cos\phi)\sin\theta/\cos\theta \quad (10)$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \quad (11)$$

$$\dot{\psi} = (q\sin\phi + r\cos\phi)/\cos\theta \quad (12)$$

The angle of attack α is related to the body velocity u (in x-axis) and w (in z-axis):

$$\tan\alpha = \frac{w}{u}$$

Differentiating both sides with respect to time, and re-arranging the equation to obtain the rate of angle of attack:

$$\dot{\alpha} = \frac{\dot{w}u - w\dot{u}}{u^2}\cos^2\alpha \quad (13)$$

The sideslip angle β has the following relationship with the body velocities:

$$\sin\beta = \frac{v}{\sqrt{v^2 + u^2 + w^2}}$$

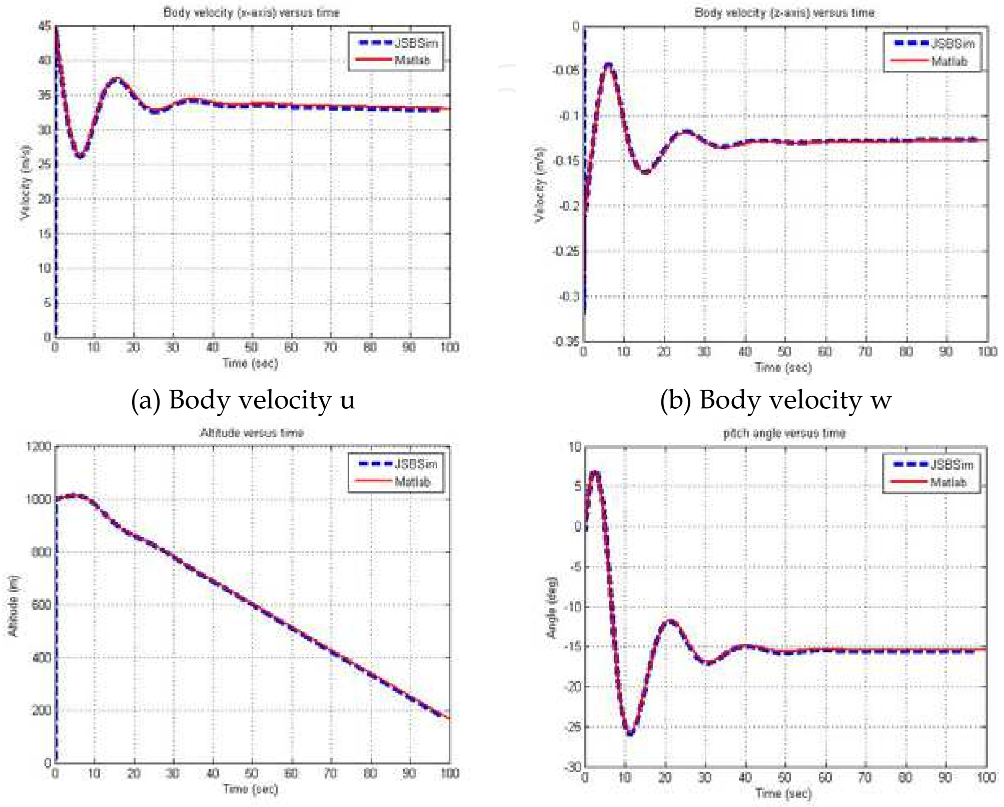
By differentiation, the rate of sideslip angle is obtained:

$$\dot{\beta} = \left[\dot{v}\left(u^2 + v^2 + w^2\right)^{-1/2} - \frac{1}{2}v\left(2u\dot{u} + 2v\dot{v} + 2w\dot{w}\right)\left(v^2 + u^2 + w^2\right)^{-3/2}\right]\frac{1}{\cos\beta} \quad (14)$$

The above ordinary differential equations (1 to 14) define a general aircraft mathematical model. The sum of forces and moments are not specified, because they vary according to aircraft geometries. Since forces and moments are obtained from aerodynamic coefficients, the summation terms are actually functions of flight conditions and aircraft state. Due to the inherent complexity of aerodynamics, the forces and moments are generally defined by lookup tables, which was the case throughout this project.

3.3 Flight Dynamics Model Verification

The FDM was verified by comparing its results with another FDM with exactly the same input information. The open source JSBSim model was chosen to make the comparison, because it is generally considered a very accurate FDM (Berndt, 2004). In the comparison, propulsion and control surface deflection angles inputs were set to be zero in both flight models. While this simplified the XML aircraft specification file required in JSBSim, it does not affect the comparison results because of the identical inputs in both FDMs. Fig. 3 shows the comparison results of body velocity u and w, altitude, and pitch angle for both FDMs.



(a) Body velocity u

(b) Body velocity w

(c) Altitude

(d) Pitch angle

Fig. 3. Verification of simulation results against JSBSim

The results verify that that the outputs produced by the developed flight dynamics model are consistent with the outputs of JSBSim. A new model was developed rather than using JSBSim because the Matlab interface is much easier to use and is more versatile than the complex JSBSim XML configuration system. The new FDM also allows rapid UAV simulation that is difficult in JSBSim.

3.4 Flight Dynamics Model Validation

To test and validate the model against real flights, an electronic system is employed for collecting all required flight data. Then the response of the real aircraft to environmental and control inputs can be compared to the response of the software model when the same input parameters are applied. This allows the flight model to be changed or fine tuned as required to prepare for reliable and representative virtual testing.

Fig. 4. shows an overview of the validation process. In the validation process, the flight simulation model requires wind data and servo motor input pulses to be recorded while the plane is in flight. Other electronic equipment includes GPS system (www.trimble.com), inertial navigation sensors (www.xbow.com) and data loggers (www.grcnz.com). This project focused on the Kuruwhengi series of UAVs, which includes the gas powered K100 with a 2.4 m wingspan and the K70 which is a 70% scaled down version.

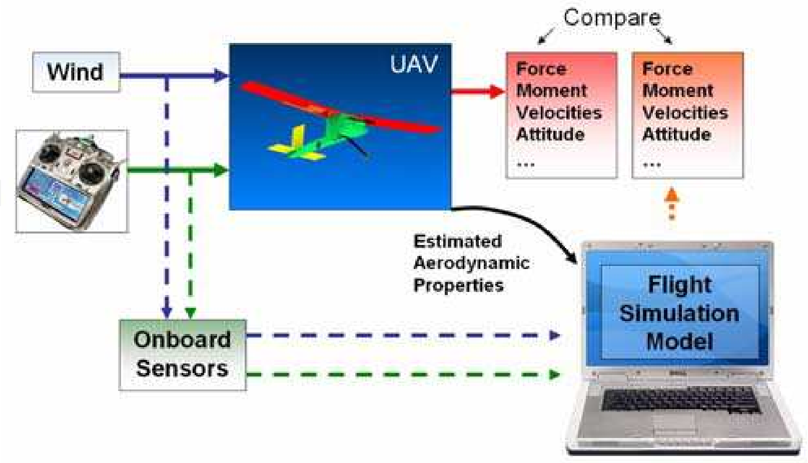


Fig. 4. Flight dynamic model validation process

The model also requires the aerodynamics coefficients and other mechanical properties of the K100 UAV. With all collected flight data from flight tests, the simulation model can be run with the same control inputs and wind conditions and the UAV behaviour compared. The major discrepancies between the two can then be identified so that changes to the simulation model for better accuracy can be made.

4. Development of Wind Speed Sensor and Data Acquisition System

The wind speed sensor was a key part of the validation project. The outcome of the project depended on the ability to measure the body velocities of the UAV in flight tests. This section presents the completion of the wind speed sensor design.

4.1 Wind Speed Sensor Requirements

The purpose of the wind speed sensor is to measure the Angle of Attack (AOA) and Angle of Sideslip (AOS) of the UAV during a flight. The data depend heavily on the wind conditions as well as the velocity and orientation of the aircraft. The requirements for the wind speed sensor design were identified as follows:
Must be light and portable to fit on the UAV.

- Must measure wind speed in three axes in order to determine angle of attack, angle of sideslip, and the longitudinal wind velocity along the body of the UAV.
- Price and complexity should be kept to a minimum.

Extensive research was carried out to determine the best method for measuring wind speed in three axes. Three main alternatives were identified and researched; namely, using ultrasound, a Pitot static system, and optical methods.

For this project the Pitot static system was chosen because of its compactness for fitting to a small aircraft and low cost. Pitot static systems work by measuring the pressure change associated with a moving volume of air. Pitot probes are used on commercial remote controlled aircraft for measuring longitudinal wind speed along the aircraft (Grasmeyer & Keennon, 2001). There are a few systems which measure angle of attack and angle of sideslip using a probe with five pressure ports at the tip (Porro, 2001; Marco, 2006). By measuring the differential pressure between opposing ports, the angle of attack, sideslip, and longitudinal wind speeds can be determined, and subsequently can be used directly to calculate a three dimensional wind vector.

4.2 Sensor System Design

The design of the wind speed sensor was split into three tasks: the mechanical design (casing), the electronics hardware, and the software design. An overview of the system is shown below in Fig. 5. The final design is a robust and compact system which is mounted in front of the leading edge of the UAV on a fibreglass rod, as shown in Fig. 6.

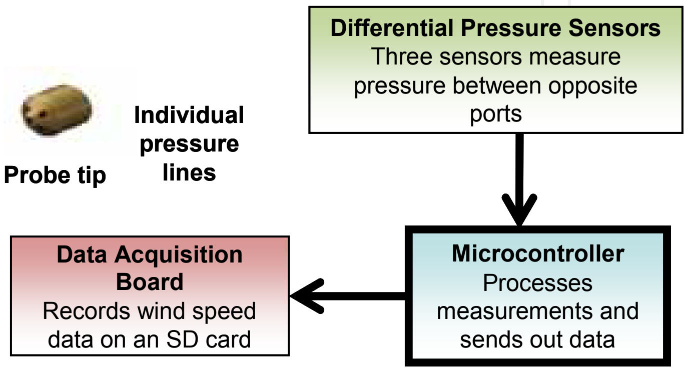

Probe tip
Individual pressure lines
Differential Pressure Sensors
Three sensors measure pressure between opposite ports
Microcontroller
Processes measurements and sends out data
Data Acquisition Board
Records wind speed data on an SD card

Fig. 5. Wind speed sensor design overview

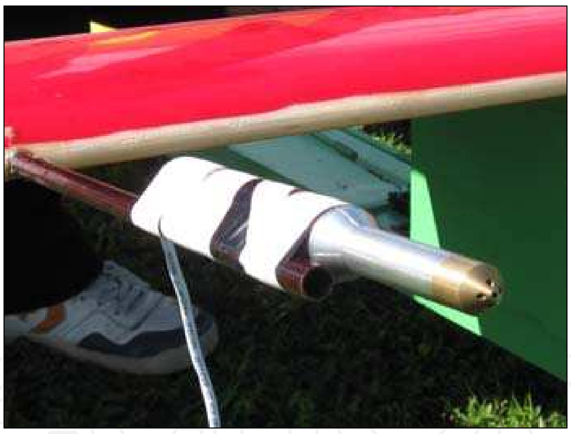


Fig. 6. Wind speed sensor mounting

4.3 Sensor Casing Design

The wind speed sensor probe casing was required to be reasonably light, and it was decided that all of the pressure sensors and electronics should be housed within the probe casing to avoid long pressure lines. The UAV could easily crash during testing, so the strength and ease of repair and assembly of the probe were also considerations. The material of the body is aluminium and the tip is made out of brass for high precision and better surface finish. The bigger diameter section of the probe body is for housing the sensor's printed circuit board (PCB). The brass probe tip is threaded onto the body so that it can be taken off the probe for ease of assembly. This also provides options to try different tip designs without changing the probe body. The tip is machined with a conical shape with a 45^0 angle to give a detecting angle range of ± 45^0. Plastic pipes connect the internal pressure sensors with the five pressure ports. The casing design shown in Fig. 7 was modelled in SolidWorks to produce drawings for fabrication and to check for the internal spacing with a PCB 3D model.

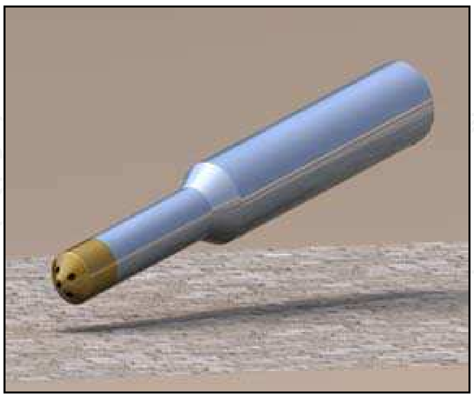


Fig. 7. Wind speed sensor casing modelling

4.4 Sensor Electronic Design

Based on first principle, the relationship between airspeed and change in pressure is given by Equation. (15). The flight speed range of a UAV is assumed to be from 30kph to 120kph, which will give a pressure difference of 42Pa and 665Pa respectively. These results showed that a sensor range from -2kPa to 2kPa is suitable for this application.

$$V = \sqrt{\frac{2RT}{M}\frac{P_{\Delta}}{P_s}} \tag{15}$$

Where V = Airspeed, R = Gas constant, T = Temperature, M = Molecular mass constant, P_{Δ} = Change in pressure, and P_s – Standard atmosphere pressure

Based on these pressure ranges, pressure sensors were chosen for the application. Three Freescale semiconductor piezoresistive pressure transducers are used (MPX7002DP); one measures the difference between vertical pressure ports (AOA), one the difference between horizontal pressure ports (AOS) and the other one the difference between the stagnant pressure in the central port and the static pressure inside the probe body (longitudinal wind velocity). These sensors have built in amplification circuitry and supply a 0 to 5V signal (which is 2.5V when the ports are at the same pressure). An Atmel ATMega 168 microcontroller receives the three analogue signals and converts the readings to a digital value by way of the built-in Analogue to Digital Converter (ADC). The digitalized values are sent out to the data acquisition board by way of a Serial Peripheral Interface (SPI). This interface is not typically used for long transmission lengths, but was tested with a 2 metres Cat-5 cable with no transmission errors. The PCB was modelled before manufacture and assembly to test the fit in the casing as shown in . Software development for the microcontroller was completed in C language using the WinAVR toolchain.

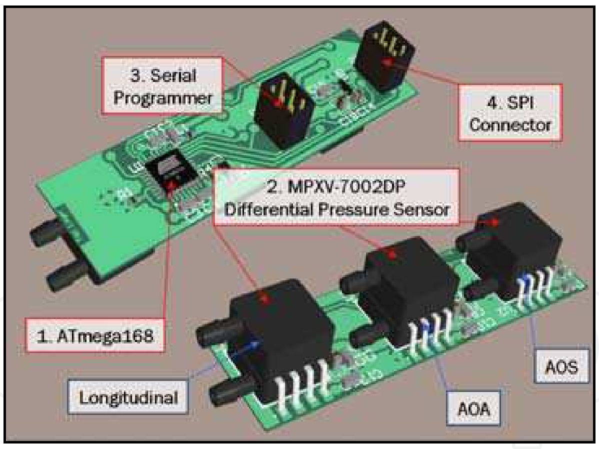


Fig. 8. Wind speed sensor PCB modelling

4.5 Sensor Calibration

The completed wind speed sensor required calibration so that the raw ADC values produced by the probe could be correlated to an actual wind vector. The closed high speed wind tunnel in the Mechanical Engineering Department at The University of Canterbury was used for the calibration. The wind speed sensor probe was mounted on a sting for accurate angle variation, as shown in . The wind tunnel was run at a range of wind speeds, and the AOA and AOS of the probe were varied and the ADC logged using a development board for a UART interface to a PC.

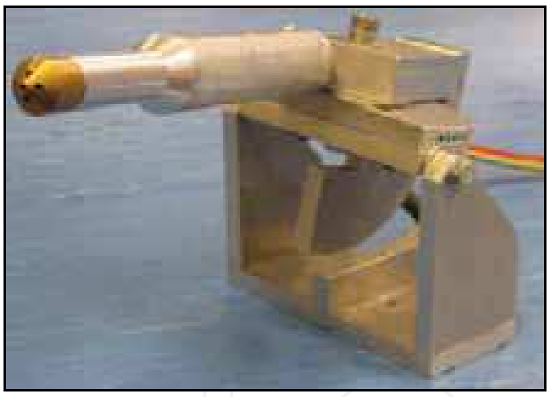


Fig. 9. Mount for wind tunnel calibration

The calibration results showed the inter-dependency of the two differential port ADC values and the central port ADC value at different speeds and angles. This is caused by the pressure drop in the central port as the angle of the probe with respect to the oncoming wind velocity increase. The inter-dependencies made generating an equation for converting ADC values into angles and velocities an extremely complex task. A sample of the calibration results for the AOA is shown in Fig. 10.

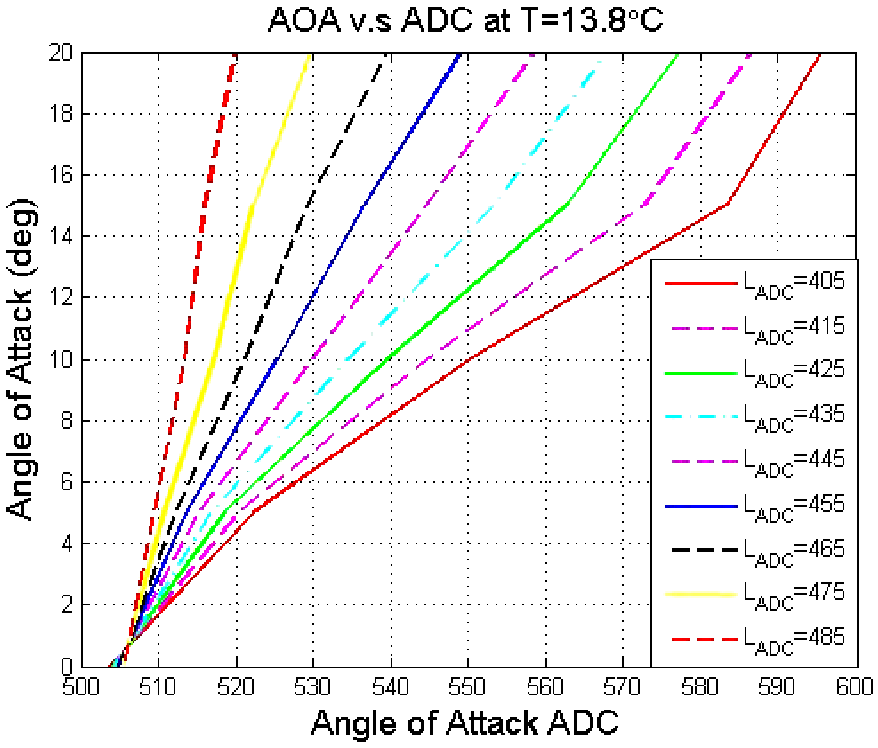


Fig. 10. Sample AOA calibration results

The calibration data were used to create a 2D look-up table for converting ADC values to angles and velocities. Matlab was used to create and use the tables, which also use a linear approximation between data points to allow reasonable estimation of all angles and velocities. The accuracy of the look-up method increases with more data points so running a calibration with as many data points as possible may increase the accuracy of the wind speed sensor significantly.

4.6 Data Acquisition Board

The Data Acquisition Board (DAB) measures the servo motor input pulse timing, which can be converted into an angle of deflection of the corresponding control surface/throttle position. The DAB also communicates with the wind speed sensor. It organizes all the data and sends them for logging by the Data Logger over a UART interface. Fig. 11 shows the functional block diagram of DAB interfacing with the other system components.

The DAB was required to achieve the following:

- Be powered off a power supply ranging from 5 to 12 volts.
- Recognize and read up to 8 servo motor channel pulses.
- Act as a gateway for the wind speed sensor and the Data Logger by communicating with the wind speed sensor via Serial Peripheral Interface (SPI) and the Data Logger via Universal Asynchronous Receiver Transmitter (UART).
- Interleave the servo signals and wind speed data; format the information and output the data via UART through an RS-232 serial interface.

Given the design requirements, the main consideration in the design process was component selection. Research was done into suitable hardware components and circuits. One circuit design consideration was whether or not to use a multiplexer to read the servo lines. The potential advantage of this approach was to save I/O pins. In addition, the software would be easier to develop because the task of interleaving the signals would be performed by hardware. The drawback was that the servos could only be connected to specific headers. This is because the RC receiver outputs control pulses one at a time to each servo sequentially. If the servo order is not specified and correctly arranged, no servo pulse may be recorded. Since only few I/O pins are required in this embedded application, it was decided to measure each servo channel directly by an I/O pin using external interrupt. This software multiplexing approach was able to deliver a more robust system.

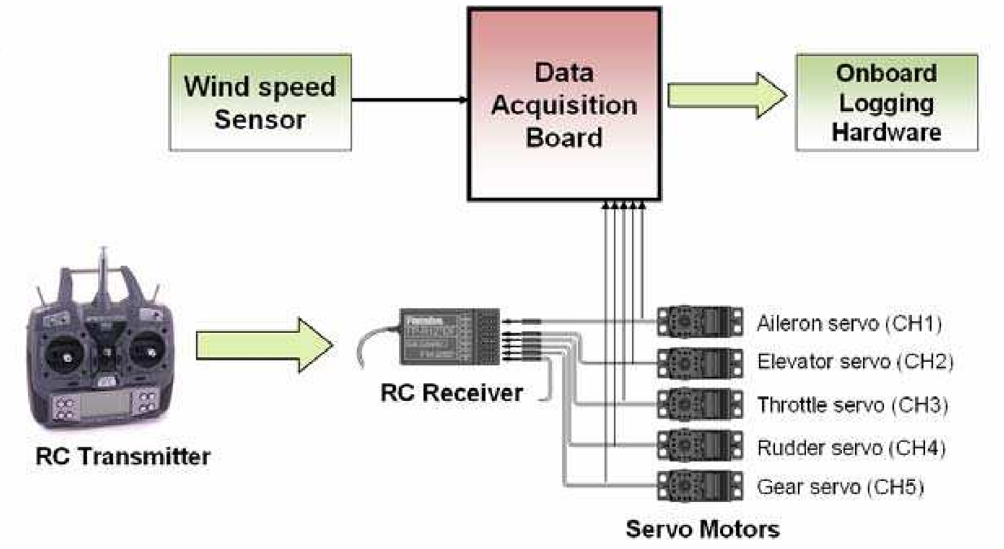


Fig. 11. Data acquisition board block diagram

The AVR ATmega168 was chosen for the main processor. This microcontroller is the same as the one used for the wind speed sensor. This simplified programming and interfacing between the two systems. The ATmega168 provides sufficient IO (input/output) and communication peripherals which are required for the Data Acquisition Board design - specifically UART and SPI.

The DAB taps off the servo control lines to read the signals. One concern was that this could affect the servo behaviour, which can potentially result in crashing the UAV. To ensure that any effect is minimal, the servo input circuitry has been designed for very high impedance. The CMOS CD4050B logic level converters from Texas Instruments were chosen to buffer the servo signals and their purpose is two-fold. Firstly, CMOS devices inherently have very high input impedance and this provides the necessary interference immunity for the servo signals. Secondly they perform the task of buffering the servo control readings to a suitable voltage level for the Atmega168 to read, allowing for a servo signal of up to 15V (while the Atmega168 is only 6V tolerant).

The final design was a success and performed to all the design specifications. It was able to reliably relay the servo signals and wind speed data via UART both during PC tests and in actual flight tests. Fig. 12. shows the final Data Acquisition Board.

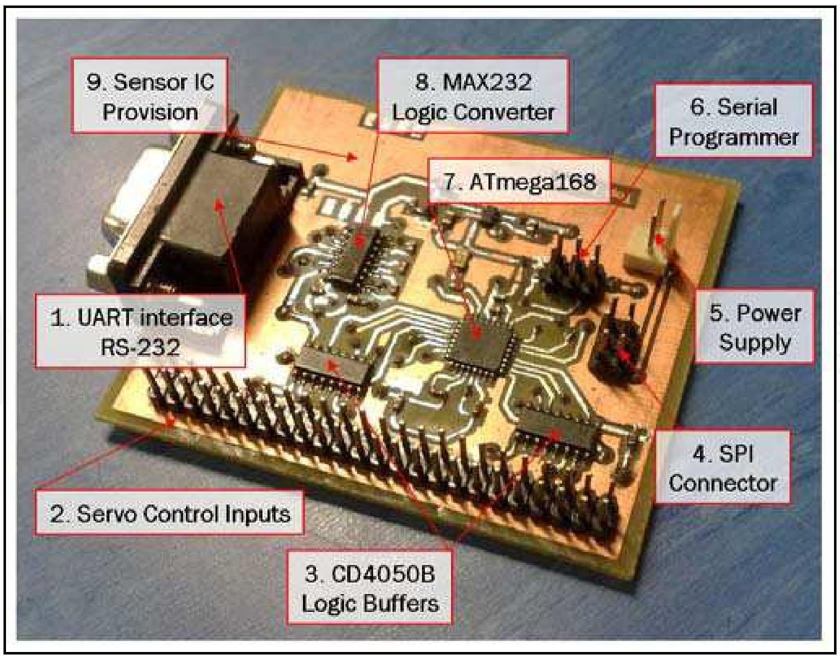


Fig. 12. Data acquisition board

A MAX-232 logic level converter was necessary to allow communication with a PC using the RS-232 standard. Although the Data Logger can handle the default logic levels of the ATmega168 IC, interfacing with a PC was required for development and testing. The MAX-232 was capable of buffering the CMOS logic levels to the necessary ±12 volt.

Noise immunity for analog pins from any possible high frequency noise was considered. The design has incorporated separate analog and digital grounds for analog and digital signals to limit the coupling between the two signals. Although there are no analog signals on the Data Acquisition Board in the current configuration, the final design incorporates footprints for the same pressure sensor IC as used in the wind speed sensor. This could provide extra functionality in development and testing. The data acquisition board is located within the body of the UAV so that the pressure sensor could be used to monitor internal 'cabin pressure', or provide a static pressure reference to compare with the static pressure in the wind speed probe.

5. Parameterisation of UAV

As discussed previously, the FDM requires the basic physical properties of the UAV to run an accurate simulation. The measurement of the K100 mechanical properties and aerodynamic coefficients is discussed in this section.

5.1 Geometric Properties of the K100 UAV

The geometry of the K100 UAV was carefully measured. The measurements were used to determine the UAV's aerodynamic coefficients using software packages. An accurate 3D model of the K100 UAV and engineering drawings were produced in SolidWorks.

The other important measurements required by the FDM are the moment of inertia along each axis, and the location of the centre of gravity. The 'compound pendulum' and 'multiple pendulums' experiments (Blaine, 1996) were carried out on the K100 to obtain the radii of gyration in each axis. These methods are commonly used on radio controlled airplanes and missiles. During the experiment, the periods of the plane swung on each axis were obtained, as well as some relevant lengths of the setup that were required for calculating the radii of gyration. Using this information as inputs to the software program Plane Geometry (Blaine, 1996), the moment of inertia along axis was determined. Table 1 summarizes the results.

	Pitch	Roll	Yaw
Period (s)	2.21	2.26	1.49
L (mm)	1027	1027	460
R (mm)	-	-	920
Radius of Gyration (m)	0.437134	0.498094	0.50368
Mass (Kg)	5	5	5
Moment of inertia (kgm2)	0.9554307	1.240488	1.26848

Table 1: Pendulum experiment results

5.2 Software Aerodynamics Coefficients

Two software packages, Datcom (Galbraith, 2004) and Tornado (www.redhammer.se/tornado) were used to estimate the aerodynamic coefficients of the K100, based on the geometric properties that were measured. Both packages perform computational fluid dynamics (CFD) calculations in order to produce their coefficient values. The CFD calculations use simplified geometry based on basic measurements only, so they do not give extremely realistic coefficient values.

Wind Tunnel Testing was required to compare the software aerodynamic coefficients with results found experimentally. Using the experimental results it was possible to determine which of the software packages produces better results for a particular coefficient and how much it differs from the experimental results. From these results it can be determined whether or not the CFD software can be used in future if aerodynamic coefficients for another UAV are to be obtained. If the software and experimental results agree well, coefficients can be accurately determined without use of a wind tunnel. Wind tunnel testing is time consuming and requires access to a wind tunnel facility, so it should be avoided if possible.

5.3 Wind Tunnel Testing

The Department of Mechanical Engineering, the University of Canterbury, has a large open wind tunnel that can be used with relatively large models. This tunnel has a 1500 mm wide nozzle that can produce 80 kph peak wind speeds. The K70 UAV (70% scale down of K100) has 1600 mm wide wings, which is slightly wider than the nozzle airflow width. Despite this, the K70 could be used in the open wind tunnel with the narrow nozzle without causing major inaccuracies. The one significant source of error when using the open wind tunnel is the limitation of its airflow speed and this had to be considered when analysing the results.

A wind tunnel test of the K70 UAV was carried out in order to determine the aerodynamic coefficients of the vehicle. These were used for inputs to the Flight Dynamic Model directly. They were also interesting for the comparison of different software packages for determining aerodynamic coefficients.

In preparation for wind tunnel testing a sting mount set up for mounting the K70 UAV was designed in SolidWorks, shown in . The mount was based on an existing sting (used in a previous wind tunnel experiment) which was fastened to a U-shaped clamp with an M12 bolt. Plates above and below where the UAV was positioned were fastened onto the clamp using M6 bolts. The UAV was clamped to the sting because it could not be drilled or modified in any other way. The sting was bolted to a force plate using a series of 3 mm OD super screws in order to minimise the damage to the force plate. The force place measured the forces and moments that the sting was subjected to during testing using a series of strain gauges. The wind tunnel setup and the UAV under test are shown in .

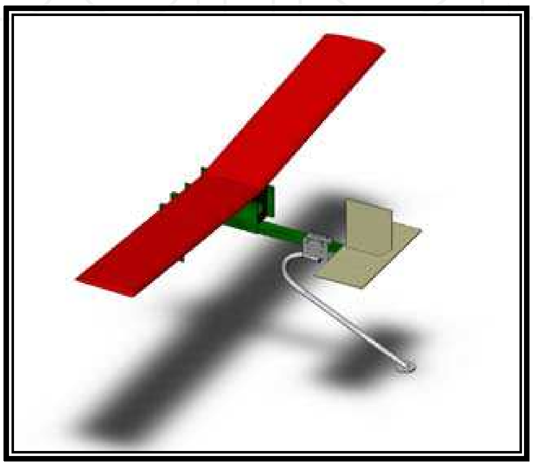


Fig. 13. SolidWorks UAV and sting mount

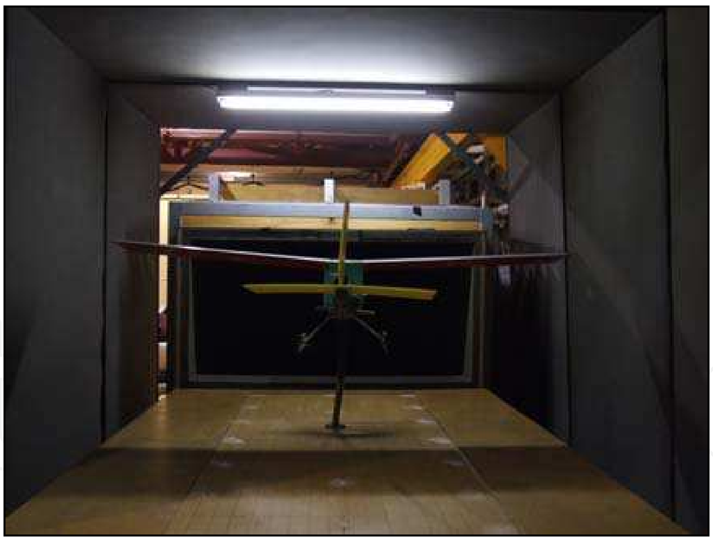


Fig. 14. Wind tunnel test setup

During Wind Tunnel testing, four load cell force readings were transmitted to LabVIEW (www.ni.com/labview) via a serial connection. LabVIEW was used to convert the raw force measurements into useful parameters – drag force, lift force, pitching moment and rolling moment. In order to be able to scale this data the software had to be calibrated. The software was reset at the start of each run. After each UAV test, known forces and moments were applied to the load cells which allowed the software to apply a scaling factor to the results. A sample readout of the LabView interface is shown in Fig. 15.

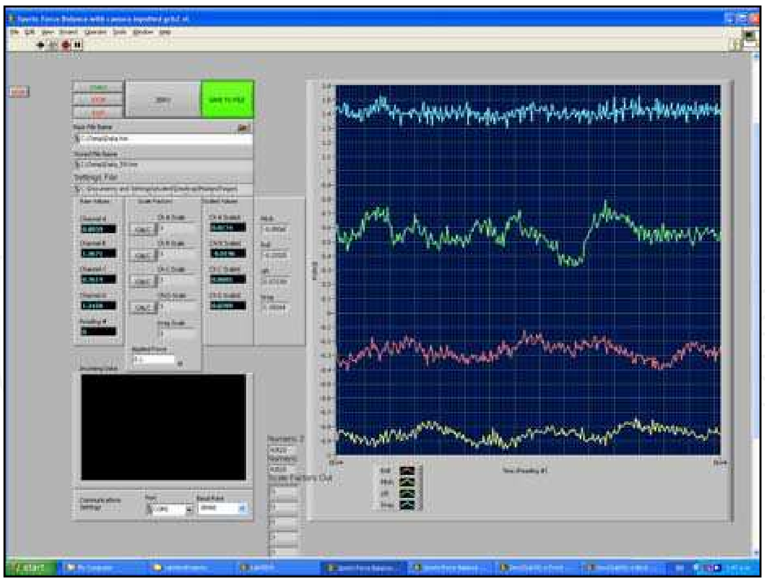


Fig. 15. LabVIEW wind tunnel output

For calibration an 18N (2 kg) weight was used. It was applied on the middle of the force balance to calibrate the lift force, pulled over the front using a pulley system to calibrate drag force, applied to the side a known distance to calibrate rolling moment (M = Fd) and applied to the front a known distance from the sting base to calibrate pitching moment. With two points (the zero value and the set value) LabVIEW can interpret the load cell data and hence calibrate itself since the relationship between force plate forces/moments and load cell transmission data is linear. Once the calibration was completed, the wind tunnel was started and the airflow was increased progressively from 0 kph to the maximum of 72 kph. The four load-cell series was plotted against time and the results exported to a spreadsheet using a LabVIEW application. The run was completed and the wind tunnel turned off once stable results were observed at maximum airflow speed. The conditions for the experiment are shown in Table 2.

Wind speed	72 ± 2kph	20 ± 0.5 ms-1
Pressure	985 ± 10 mbar	0.98 ± 0.01 atm
Temperature	16 ± 2 °C	289 ± K

Table 2. Experimental conditions

Runs were repeated for pitching and yawing angles -30 to 30 degrees inclusive with five degrees increments. In reality only pitching angles -10 to 20 degrees need to be considered because a plane will naturally stall outside this range (but the testing was done to show values outside this range nonetheless). Smaller increments of two degrees would have been optimal but because of mounting difficulties and error in the angle setup this was not easily achievable. The error in the angle setup was due to the UAV being mounted at a position away from its centre of gravity, at the back of fuselage. The UAV typically sagged forward and this could not be avoided because of mounting limitations (the mount could not be viced or clamped any more firmly without causing damage to the UAV) and flexing of the UAV airframe. As a result, during runs the UAV had some vibration which caused oscillations in the LabVIEW output. Therefore average values were used as opposed to maximum values.

5.4 Wind Tunnel and Software Coefficients Comparison

The LabVIEW experimental output was post-processed using Microsoft Excel and Matlab. The Lift Force, Drag Force, Pitching Moment and Rolling Moment were recorded for each trial. With images taken before each test the setup angle (pitching/yawing) could be analysed and the frontal area subjected to the airflow could be calculated using a pixel counting technique. With all of this data experimental coefficients were produced according to the following formulae:

$$C_x = \frac{F_x}{\overline{q}A} \tag{16}$$

$$C_x = \frac{M_x}{\overline{q}A} \tag{17}$$

$$\overline{q} = \frac{pV^2}{2} \tag{18}$$

where C = Aerodynamic Coefficient, F = Force (e.g. Lift force for coefficient of Lift), M = Moment (e.g. Pitching moment for Pitching coefficient), $\overline{q}$ = Dynamic pressure, A = Frontal area, p = Air density, V = Air velocity. (Since a 70% scaled model K70 was used, it is divided by the scale factor 0.7)

These results show that the existing software values are similar to what was determined using software with the exception of air drag. The reason the experimental drag coefficient is much greater than its software equivalents for the whole angle range is due to the software packages geometry limitations. Fig. 16 compares the coefficient profiles produced by the software packages and the wind tunnel results, and Fig. 17 shows how the coefficients change with yawing angle. It can be clearly seen that the experimental values display much more drag. For typical model aircraft the Datcom and Tornado drag coefficient estimates may be reasonably accurate, but for the K100, with its untypical blunt shape and large frontal area, the drag is obviously going to be much higher.

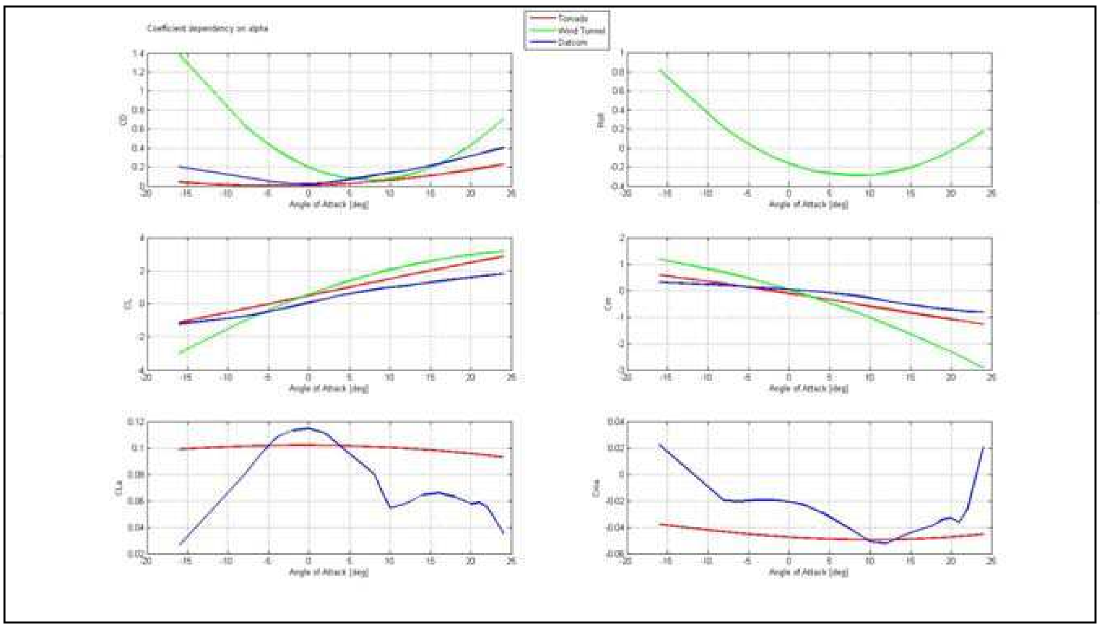


Fig. 16. Comparison of computed and experimental coefficients

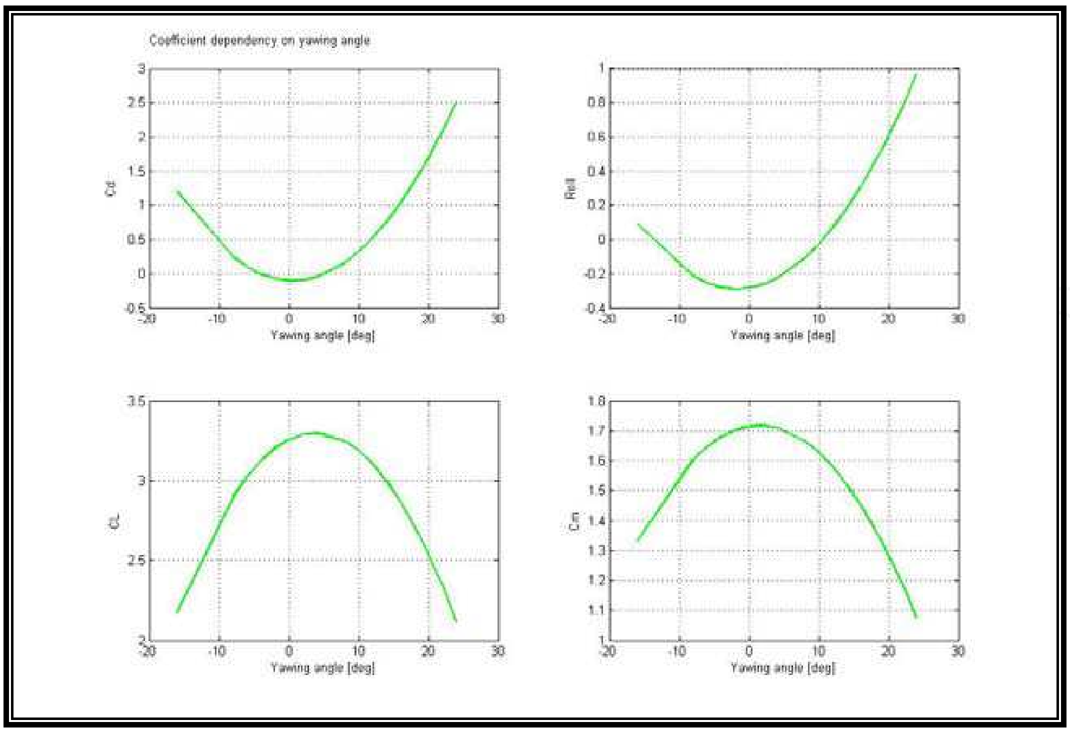


Fig. 17. Experimental coefficients for yawing angle changes

By substituting the software coefficients with the newly found experimental values, the wind speed velocity error observed in a flight simulation using the FDM is reduced and the flight dynamics model flight path improves. This verifies that the flight dynamics model can simulate a flight path similar to an actual flight given the control inputs are reasonably accurate. If the coefficients could be determined even more accurately the model may improve further.

6. Experimental Validation and Discussion

6.1 Flight Test Data

After careful preparation and organization, the flight tests using the model aeroplane K100 UAV were conducted. The flight test required the following field apparatus:

- K100 UAV – Perform flight tests, and record data.
- GPS Ground Station – Provide a stationary GPS reference.
- Camcorder – Record video of flights.
- Laptop – Perform data analysis in field.

Before a flight could be performed the GPS Ground station and Camcorder were set up. To ensure the data would be representative of a wide range of plane behaviour, it was necessary to gather data for all basic plane manoeuvres such as taxiing, take off, in flight movement and landing. Each flight lasted no less than five minutes. The data was downloaded to a laptop for analysis in between each flight to determine if it was useable.

The K100 was flown a number of times. The flight tests resulted in a 5 minute flight with good flight data for all of the parameters that required measurement. A considerable amount of post processing of data was undertaken for the flight test. All of the flight data that was logged in SD memory cards was needed to be interpreted in some way before any useful conclusions could be made. The attitude and position data were processed by combining information collected from the GPS base station and the onboard GPS and AHRS (www.xbow.com). The static position data provided by the GPS base station determined the errors of GPS signal. More precise navigation information can be obtained by subtracting the known errors.

Three ADC values were collected by the wind speed sensor in real time: differential pressure on angle of attack and sideslip, and the stagnation pressure in the central port. Even at a low sampling rate of 10Hz, the data captured by the wind speed sensor appeared to be quite noisy. By applying interpolation on the 2D lookup tables that were obtained in wind tunnel calibration, angle of attack α, angle of sideslip β, and body velocity along the x-axis u were derived. The other two body velocities v and w were easily derived using simple trigonometric relationships:

$$v = u \cdot tan(\beta) \tag{19}$$

$$w = u \cdot tan(a) \tag{20}$$

Since the flight model only accepts inputs for control surfaces in terms of deflection angles, conversions from servo pulse timing to the corresponding control surface deflection angles had to be made. Likewise, the thrust produced by the engine had to be correlated to the throttle servo pulses. The relationship between pulse servo signal pulse widths versus deflection angles and thrust were measured on the K100. The measurements are shown in Table 3 to Table 6. Interpolation was used to convert servo pulse widths into the mechanical inputs of the plane.

Servo Pulses (ms)	0.9	1.255	1.426	1.73	2.1
angle (deg)	16	8	0	-0.6	-1.3

Table 3: Elevator

Servo Pulses (ms)	0.9	1.125	1.25	1.45	1.565	1.60	1.78	1.9	2.1
Angle (deg)	18	14.4	10.8	6	0	-6	-10.8	14.4	-18

Table 4: Ailerons

Servo Pulses (ms)	0.8	0.97	1.33	1.544	1.72	1.91	2.1
Angle (deg)	12	10	5	0	-5	-10	-12

Table 5: Rudder

Servo Pulses (ms)	0.8	0.95	1.13	1.30	1.5	1.7	1.99	2.1
Thrust (N)	36.3	35.3	32.37	31.39	24.52	19.62	11.77	9.81

Table 6: Thrust

Fig. 18. and Fig. 19. show the servo pulse width signals and the converted control surface deflection angles and thrust. The collected signals covered the whole flight testing from taking off and landing. These results showed consistency with the UAV behaviour observed by watching the video recorded during the flight test.

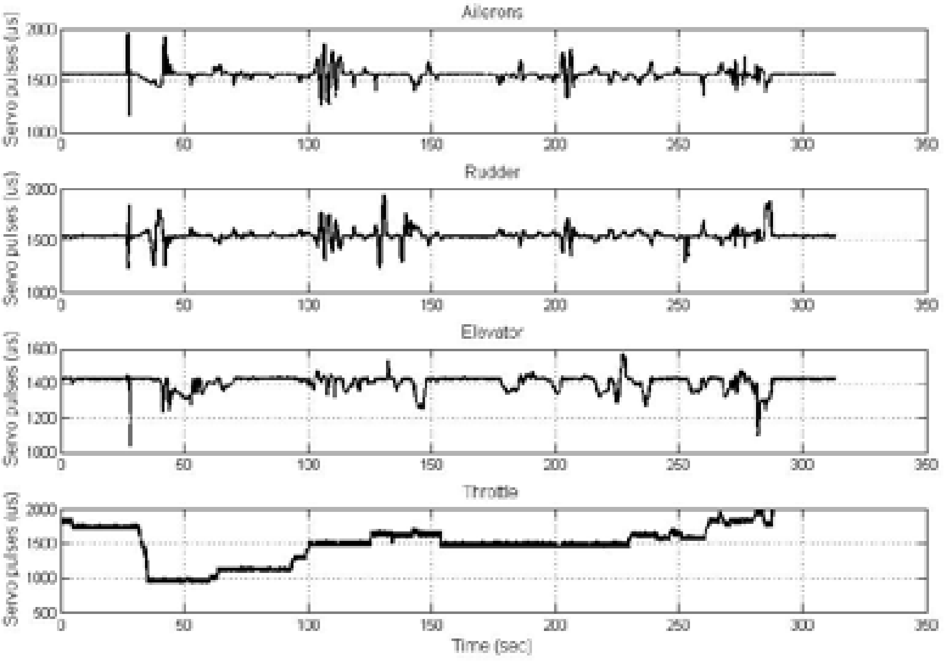


Fig. 18. Servo input signals

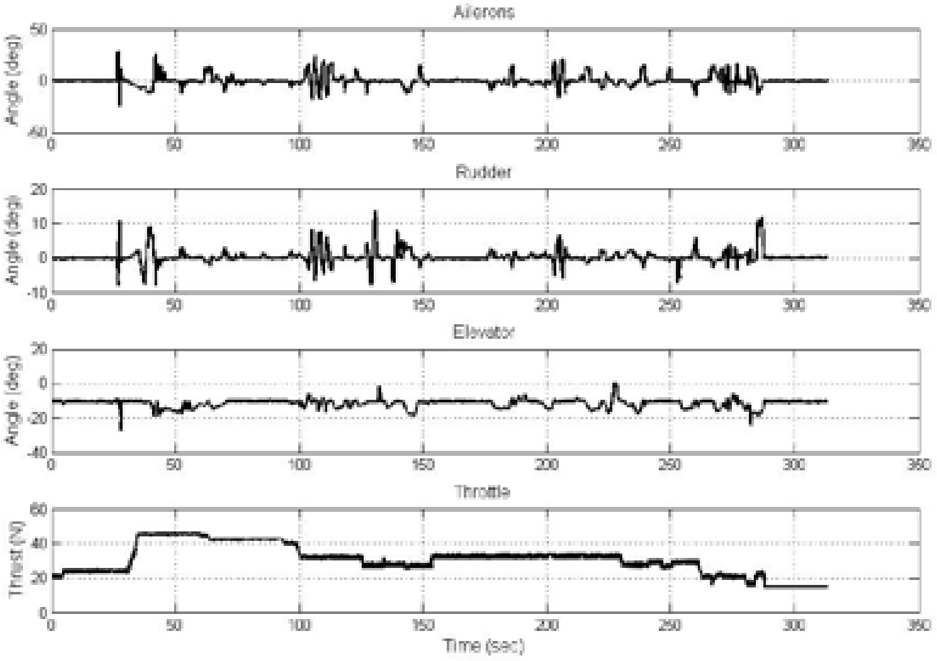


Fig. 19. Deflection angles and thrust

The vibrations from the two-stroke engine of the K100 caused problems for the wind speed sensor, which can be seen in Fig. 20. The vibrations of the sensor caused erroneous detection of angle changes, especially when the plane was stationary or moving slowly. This was caused by the probe tip movements resulting in a small pressure to be induced between opposite ports on the probe. Because of the slow speed, the longitudinal reading was low.

This condition is normally only present at very large angles. This theory was confirmed by the observation that only the AOA data was affected by this phenomenon. The sensor could only vibrate in the vertical direction because of how it was fixed on the plane wing. Note the UAV was stationary for the first 35 seconds of data recording. Vibration remained in 100 seconds after the throttle was turned down as can be seen in Fig. 20.

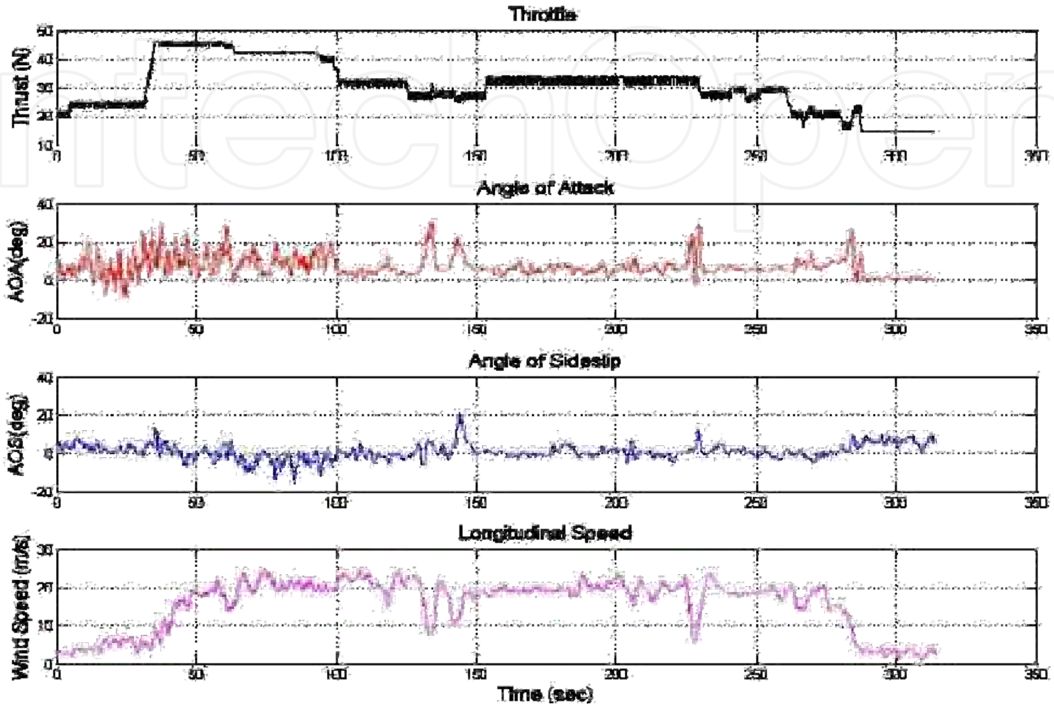


Fig. 20. Wind speed sensor data

6.2 Validation Results

Taking off and landing are generally much more difficult to model because of the more complicated environment. The added intricacies can be introduced by ground effect, low wind turbulence and the high non-linear flight response at low speed. As a result, only a section of the inputs were used to validate the flight dynamics model. Referring to Fig. 20, the chosen section for simulation was the period from 100 to 220 seconds. Fig. 21. shows the simulation results that used aerodynamic coefficients from Datcom exclusively. They were compared to the aeroplane responses that were measured by the onboard inertial reference system. The comparison was based on aeroplane attitude, altitude, flight path and body velocities. The roll angle, altitude change, and the vertical axis body velocity *w* generated by the FDM agree well with the actual flight response. In particular, the simulated roll angle gave the best match to the measured response. The simulated body velocity along x-axis *u* shows less resemblance to the experimental results. The much higher speed obtained from the model indicated that the drag coefficient used by the model is lower than the actual coefficient. The flight path is related to the integral of the velocities, so that both of flight path and body velocities exhibit a similar degree of inaccuracy.

The determined drag coefficient from the wind tunnel testing was used to replace the coefficient determined by Datcom. The simulation results given in Fig. 22. show a significant improvement for the body velocity parameters. The body velocity *u* shows that the flight model was not able to predict the velocity changes that occurred at around 30 and 40 second into the flight sample, but using the experimental coefficient for drag removed the offset error that can be seen in Fig. 22. In addition, the pitch angle was matched slightly better to the actual response. However, relatively large errors still exist on yaw angle and the flight path. The remaining experimental aerodynamic coefficients were not used for simulation because they produced an unstable flight response when used for the simulation.

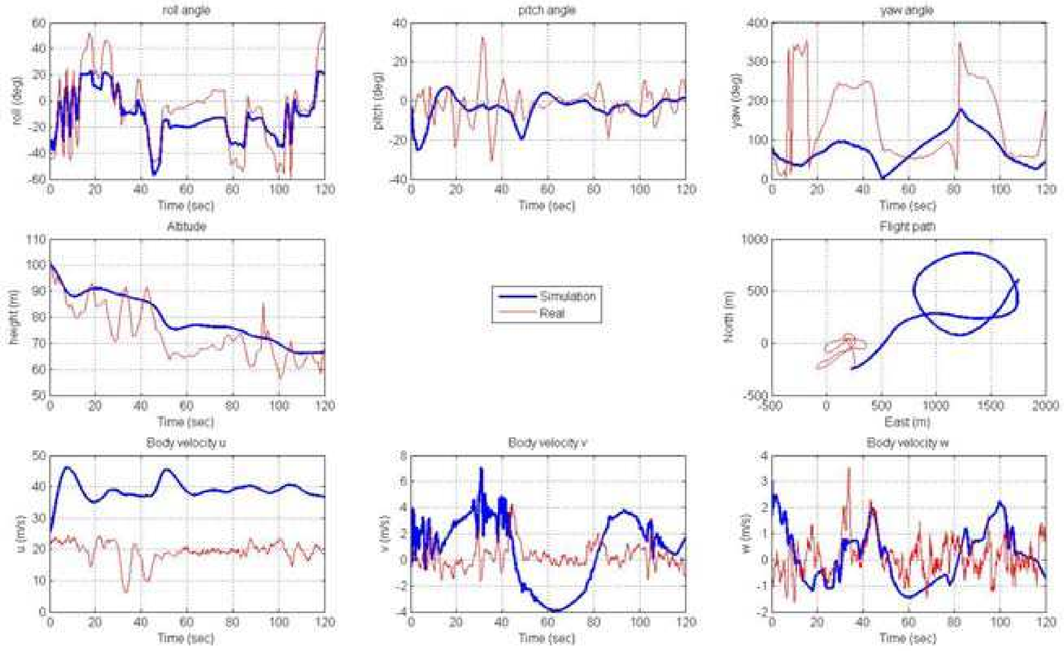


Fig. 21. Results based on coefficients generated by Datcom.

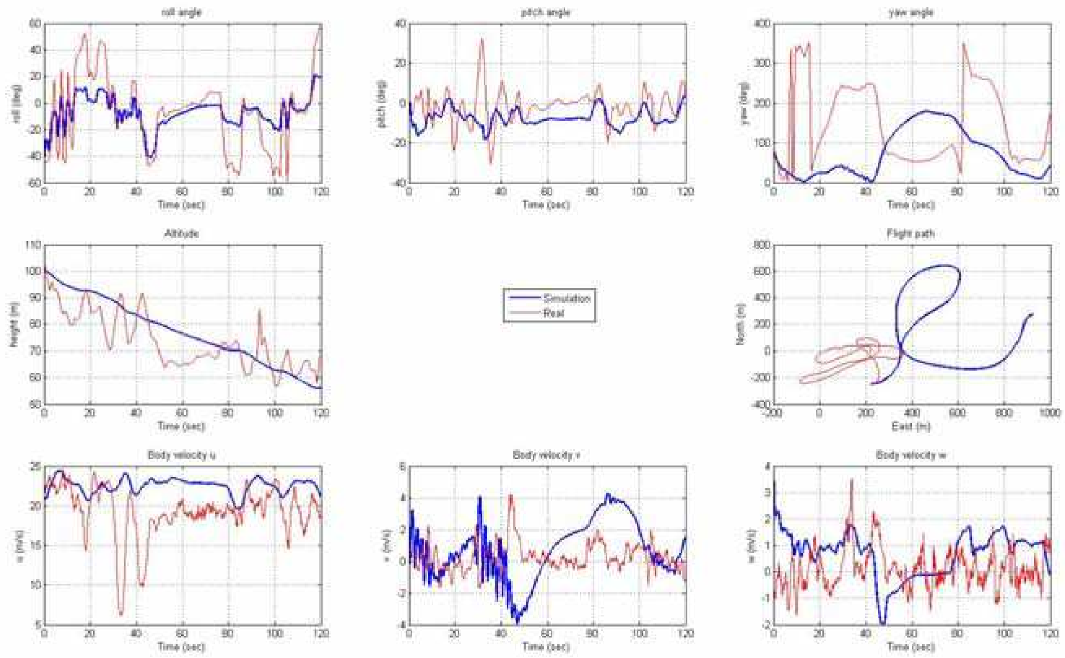


Fig. 22. Simulation results with wind tunnel determined drag coefficient

6.3 Model Errors and Improvements

The validation process has revealed the reliability of the simulation results given by the flight dynamics model with all the determined inputs. The modelling process is complex because of the variety of data that must be collected. There are many factors that can affect the simulated behaviour of the aircraft. Even though the simulation was able to predict the general trend of the aircraft motion, the large error found on the flight path and longitudinal velocity has limited its use for the application of dead-reckoning. The major sources of errors likely are as follows.

Control surfaces have a profound effect on the response of the aircraft. These effects are governed by the control surface aerodynamic coefficients, and they are normally non-linear and heavily dependent on the aircraft geometries. These coefficients are currently derived from Datcom, which determines their values from first principles. The nonlinearity of these coefficients implies that some errors must be involved from mathematical estimations.

Conversion between servo pulses and deflection angles were based on measurements taken when the aircraft was stationary. In flight, all control surfaces are subjected to high wind speed, which causes deflection and distortion. There is no easy way to measure the actual deflection angles. The angles were manually adjusted slightly to reduce this error.

Simulation errors were quantified by the measured aircraft responses. The measured responses involve uncertainties themselves due to noise, sensor limits and conversion inaccuracies. Quantifying the error in flight data instrumentation would allow an estimate of the effects of these errors on the simulation results.

Initial conditions affect the solution of a dynamic system. All initial conditions including linear and angular velocities, acceleration and position were measured by the onboard inertial navigation sensor system. This system has its own inaccuracies mostly caused by drift, which may have contributed to the flight model and experimental data discrepancies.

Wind condition inputs cause singularities when used in the current implementation of the flight model. The rapid change in wind data results in the model refusing to continue the calculation. This failure is caused by a combination of the limited quality of the wind speed data and a limitation of the model. A possible improvement on the flight model is to find out the cause and solution to this problem so that the model can include the measured wind vector in the calculation of the body velocities of the UAV. By doing so, the accuracy of the model would be improved significantly because ambient wind conditions can be taken into account. In addition, a higher sample rate in wind speed data collection would reduce the rapid rate of change in the data which causes the FDM to crash.

Effects of the control surfaces on the aircraft motion are significant. Determining the aircraft response with its control surfaces in a wind tunnel would greatly improve the simulation results.

In terms of the validation process, it was noted that a gas powered UAV can produce considerable vibrations, especially in the takeoff phase. These vibrations together with the turbulence behind the propeller caused significant noise to the wind speed sensor. This suggests a review of the mount position of the wind speed sensor and the selection of the UAV. A better position to place the wind speed sensor would be at the tip of the wing. With this position a counter weight has to be put on the other wing to cancel out the force and moment induced on the plane. An electric powered UAV would help to stop the wind speed sensor noise.

7. Conclusions

The objective of this project was to validate a flight dynamics model for the K100 UAV. This objective was achieved by determining the major aerodynamic coefficients of the K100 UAV and producing hardware for collecting flight data.

The wind tunnel testing was performed in a low speed wind tunnel using the K70 so accuracy was slightly compromised. However, the results were sufficient to show that for the unusually shaped K100 UAV, the aerodynamics coefficients determined by software packages (Datcom and Tornado) do not accurately represent the actual values. The experimental drag coefficients are higher than those predicted by the software model and this has a large affect on the accuracy of the flight dynamic model.

The sensor hardware developed during this project worked well during flight tests and allowed the collection of flight data which were used to assess the accuracy of the flight dynamics model. These sensors may be useful in other applications, such as aids for UAV navigation. The sources of instrumentation error were identified. The serious vibration generated by the K100 engines caused false AOA readings, particularly at low speeds. This could be overcome by improving the probe mounting location and method or using an electrically powered UAV.

The validation of the current software FDM has shown that it has two main limitations. It is unable to use some of the experimental aerodynamics coefficients because they produced unstable flight response. It was also unable to use the collected wind data because rapid changes caused the FDM to crash. Resolving these problems would improve the FDM, which otherwise represents the UAV flight reasonably well.

The aims of the model validation were met and a complete validation process for a flight dynamics model was presented. The current FDM has been assessed using this method and possible sources of inaccuracies identified. The presented validation process based on in-flight test and onboard instrumentation makes a significant step towards completing an accurate flight simulation system for auto-pilot development and design verification of UAVs.

8. Acknowledgement

The authors would like to thank Graeme Harris for his help with the wind tunnel testing, and Barry Lennox for his help with the flight test preparation and piloting. We would also like to thank the Geospatial Research Centre (NZ) Limited for their support throughout the project.

9. References

Barr, J. (2006). "FlightGear takes off", website: www.flightgear.org/.

Berndt, J. S. (2004). "JSBsim: An Open Source Flight Dynamics Model in C++." *AIAA Modeling and Simulation Technologies Conference and Exhibit AIAA* 2004-4923(AIAA).

Blaine K, B.-R. (1996). *Plane Geometry - Aircraft Geometry Measurement and Design Programs*.

Buschmann, M., Bange, J., Vörsmann, P. (2004). "MMAV - A Miniature Unmanned Aerial Vehicle (MINI-UAV) for Meteorological Purposes." *16th Symposium on Boundary Layers and Turbulence, American Meteorological Society*: Paper 6.7, 7pp.

Chavez, F. R., Bernard, J., et al. (2001). "Advancing the State of the Art in Flight Simulation via the Use of Synthetic Environments." *Iowa Space Grant Consortium*.

Cook, M. V. (2007). Flight dynamics principles. Amsterdam.

Cooke, J. M., Zyda, M. J., Pratt, D.R., McGhee, R.B. (1994). "NPSNET: Fight Simulation Dynamic Modeling using Quaternions." *In Presence 1*, No. 4: 404 - 420.

Diebel, J. (2006). "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors." (*Stanford University, Palo Alto, CA*).

Galbraith, B. (2004). "Datcom Predicted Aerodynamic Model." Holy Cows, www.holycows.net.

Grasmeyer, J.M., and Keennon, M. T. (2001). "Development of the Black Widow Micro Air Vehicle." *39th AIAA Aerospace Sciences Meeting and Exhibit* .

Guglieri, G., Pralio, B., Quagliotti, F. (2006). "Flight Control System Design for a Micro Aerial Vehicle." *Aircraft Engineering and Aerospace Technology* 78(2): 87 - 97.

Jackowski, J., Boothe, K., Albertani, R., Lind, R., Lfju, P. (2004). "Modeling the Flight Dynamics of a Micro Air Vehicle." *European Micro Air Vehicle Conference*.

Jordan, T. L., J. V. Foster, et al. (2006). "AirSTAR: A UAV Platform for Flight Dynamics and Control System." *NASA Langley Research Center,* Report Number: AIAA Paper 2006-3307: 8.

Lyshevski, S. E. (1997). "State-Space Identification of Nonlinear Flight Dynamics." *Proceedings of the 1997 IEEE International Conference on Control Applications*.

Marco, A. D. (2006). "A 6DoF Simulation Laboratory at the University of Naples." *The quarterly newsletter for JSBSim, an open source flight dynamics model in* C++ Volume 3(Issue 1).

Porro, A. R. (2001). "Pressure Probe Designs for Dynamic Pressure Measurements in a Supersonic Flow Field." *Glenn Research Center, Cleveland, Ohio*.

Ou, Q., Chen, X. C., Park, D., Marburg, A., Pinchin, J. (2008). "Integrated Flight Dynamics Modelling for Unmanned Aerial Vehicles." *Proc of the Fourth IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA08)*, ISBN: 978-1-4244-2368-2, Beijing, China, October 12-15, pp. 570-575.

Rasmussen, S. J., and Chandler, P. R. (2002). "Unmanned aerial vehicles: MultiUAV: a multiple UAV simulation for investigation of cooperative control." *Proceedings of the 34th conference on Winter simulation: exploring new frontiers. San Diego, California, Winter Simulation Conference*, pp. 869-877.

Ye, Z., Bhattacharya, P., Mohamadia, H., Majlesein, H., Ye, Y. (2006). “Equational Dynamic Modeling and Adaptive Control of UAV.” *Proceedings of the 2006 IEEE/SMC International Conference on System of Systems Engineering*, Los Angeles, CA, USA, April 2006, pp. 339 – 343.

Fax: +385 (51) 686 166
www.intechopen.com

Fax: +86-21-62489821