Modelling and Control of the Parrot AR.Drone

Gerard Martin

The UAV Laboratory at UNSW Canberra has recently extended their fleet to include a Parrot AR.Drone quadrotor. It is the intent of the UAV Laboratory to extend their vision-based autonomous navigation research to a quadrotor platform. Before this can be achieved the platform must be analysed and characterised. In this project a model of the AR.Drone was constructed in Simulink and validated for the purposes of supporting research within the UAV Laboratory. This project also established a fundamental body of work on the subject of quadrotors so as to enable further research by university staff and students in years to come.

Contents

I. Introduction ................................................................................................................... 1
   A. Research and Applications ....................................................................................... 1
   B. Motivation .................................................................................................................. 1
   C. Scope and Overview ................................................................................................. 1

II. Project Management ................................................................................................... 2

III. Mathematical Model .................................................................................................. 2
   A. Non-linear Model ........................................................................................................ 2
   B. Linearised Model ....................................................................................................... 3
   C. State-Space Representation ....................................................................................... 4
   D. Control Architecture and Controllers ...................................................................... 4
   E. Attitude and Position Control .................................................................................. 4
   F. Feedback Loop Delay .............................................................................................. 5

IV. Ascending Technologies Pelican Quadrotor .............................................................. 5

V. Parrot AR.Drone Quadrotor ....................................................................................... 5
   A. Overview .................................................................................................................... 5
   B. Control Inputs ............................................................................................................ 5
   C. Hardware and Software ........................................................................................... 6
   D. The AR.Drone as a Research Platform ................................................................... 6

VI. System Identification .................................................................................................. 6
   A. Software Development Kit ...................................................................................... 7
   B. Flight Tests ............................................................................................................... 7
   C. Primary Effects of Control ....................................................................................... 7
   D. Secondary Effects of Control .................................................................................. 8
   E. Other Characteristics ............................................................................................... 9
   F. Controller Testing ..................................................................................................... 9

VII. Conclusion .................................................................................................................. 9

VIII. Recommendations .................................................................................................. 10

IX. Acknowledgements ................................................................................................... 10

X. References .................................................................................................................. 11

1SBLT, RAN, School of Engineering and Information Technology, ZEIT4500/4501

Final Project Report 2012, SEIT, UNSW Canberra
Nomenclature

\{B\} = Body-fixed frame of reference (right-hand 3D Cartesian axes).
\{W\} = Inertial (world) frame of reference (right-hand 3D Cartesian axes).
R = Rotational transformation matrix to transform \{B\} to \{W\}
k_f = Rotor force constant relating \( F_i \) to \( \omega_i \)
k_m = Rotor moment constant relating \( M_i \) to \( \omega_i^2 \)
k_a = Motor response constant relating \( \dot{\omega}_i \) to \( \omega_i^{des} - \omega_i \)

\( x_B, y_B, z_B \) (m) = Position along x, y and z axis respectively in body-fixed frame, \{B\}
x, y, z (m) = Position along x, y and z axis respectively in inertial frame, \{W\}
r (m) = Position vector which points from the origin of \{W\} to \{B\}
L (m) = Distance between CoM and centre of rotor (same for all rotors)

\( \dot{x}, \dot{y}, \dot{z} \) (m \cdot s^{-1}) = Velocity along x, y and z axis respectively in inertial frame, \{W\}
\( \ddot{x}, \ddot{y}, \ddot{z} \) (m \cdot s^{-2}) = Acceleration along x, y and z axis respectively in inertial frame, \{W\}

\( \phi, \theta, \psi \) (rad) = Roll, pitch and yaw angle respectively of quadrotor (Z-X-Y Euler angles)
p, q, r (rad \cdot s^{-1}) = Roll, pitch and yaw rate respectively of quadrotor in \{B\}
\( \omega_i \) (rad \cdot s^{-1}) = Rotation speed of the i-th rotor, \( \omega_i^{des} \) is desired rotor speed
\( \omega_a \) (rad \cdot s^{-1}) = Rotation speed of rotors required to maintain hover
\( \ddot{p}, \ddot{q}, \ddot{r} \) (rad \cdot s^{-2}) = Angular acceleration

\( F_i \) (N) = Force in \( \hat{z}_B \) direction due to the i-th rotor
\( g \) (N) = Force due to gravity
\( M_i \) (N \cdot m) = Moment in \( \hat{z}_B \) direction due to the i-th rotor
\( \tau \) (N \cdot m) = Torque
m (kg) = Mass of quadrotor
I (kg \cdot m^2) = Inertia tensor

Abbreviations

ABL Allocated Baseline (Systems Engineering Document)
ADC Analog to Digital Converter
ADFA Australian Defence Force Academy
AoA Angle Of Attack (Angle of wing/rotor to air flow)
CoM Centre of Mass (Equal to the centre of gravity (CoG) in uniform gravitational fields)
COTS Commercial Off The Shelf (An acquisition option)
DCT Discrete Cosine Transform
FBL Functional Baseline (Systems Engineering Document)
IAW In Accordance With
INS Inertial Navigation System (Accelerometers and Gyroscopes)
IOT In Order To
ITR Initial Thesis Report
LiP Lithium-ion Polymer (Type of High-Capacity Battery)
LQR Linear Quadratic Regulator (Type of Controller)
MEMS Microelectromechanical System
MIMO Multiple Input, Multiple Output (Type of System)
PID Proportional Integral Derivative (Type of Controller)
ROS Real-time Operating System
School SEIT – School of Engineering and Information Technology
SISO Single Input, Single Output (Type of System)
SLAM Simultaneous Localisation and Mapping
SRD Stakeholder Requirements Document (Systems Engineering Document)
UNSW University of New South Wales
VTOL Vertical Take-Off and Landing
I. INTRODUCTION

A. Research and Applications

The concept of the quadrotor is not new; the first quadrotor, a manned rotorcraft named Gyroplane No. 1, took flight in 1907 [1]. Recently however, advances in INSs using MEMS technology, the availability of high speed brushless motors and high power-to-weight LiP batteries [2] has meant small, unmanned quadrotors can now be developed. With adequate control, quadrotors are capable of VTOL, aggressive manoeuvres [3], precise manoeuvring in confined spaces and formation flying with multiple quadrotors.

Many universities and research institutions are currently conducting research into quadrotors, most notably the GRASP Lab at the University of Pennsylvania [4] who have produced many papers, have many videos of aggressive quadrotor manoeuvres online [5] and have presented at TED2012 [6]. While research is still being conducted, there are already quadrotors in the commercial market. Remote control quadrotor toys are currently available such as the Parrot AR.Drone [7] (see Figure 1) which can be controlled via smartphone and quadrotors, such as those from DraganFlyer [8], are being used in industry for tasks such as plant inspections, and by police and fire departments [9], for dangerous and monotonous tasks.

B. Motivation

The UAV Laboratory at UNSW Canberra conducts research into vision-based autonomous navigation and currently implements controllers on traditional form helicopters. The Laboratory intends to extend their research to quadrotor platforms and hence requires an analysis and model of their recent acquisitions: the AscTec Pelican and the Parrot AR.Drone. This project established the required body of knowledge and model of the Parrot AR.Drone in order to support the research activities at the UNSW Canberra UAV Laboratory.

Quadrotors have a number of advantages over traditional helicopters due to their mechanical simplicity as there is no requirement for a swash plate and varying rotor AoA. Design and maintenance is simplified as a result. Additionally the smaller diameter rotors are safer for human interaction due to a lower kinetic energy.

C. Scope and Overview

This aim of this project was to both establish a fundamental body of knowledge on the subject of quadrotors and perform a system identification on the Parrot AR.Drone. In this regard a balance was required between the broad knowledge requirement of general quadrotor research and the depth of knowledge requirement for the device research and system identification process. The scopes of individual aspects of the project were hence an important consideration especially since limited time was available. The following areas of research were identified as essential to the project and hence defined the scope:

1) Literature Review – a literature review was conducted so as to understand the research currently being conducted, the applications of the research, and importantly any research immediately applicable to the projects main goal of system identification.
2) Dynamic Model – a dynamic model was derived from first principles so as to understand the structure of the system and possible control issues. Self-education on 6DOF dynamic modelling was hence required.
3) Linearised Model – the non-linear model was linearised about the hover point so as to enable traditional linear analysis techniques. At attitudes less than 30° the linearised model is sufficiently accurate.
4) Control Testing – a control architecture was tested with PID and PD controllers to validate the use of both. It was determined that higher order controllers such as LQR controllers were not necessary for this application and so effort was instead directed to understanding the effect of the $k_p$, $k_i$ and $k_d$ gain constants.
5) Ascending Technologies Pelican Quadrotor Device Research – preliminary research was conducted into the AscTec Pelican however it was soon discovered that the device was not suitable. IAW the ITR management plan the Parrot AR.Drone was investigated instead.
6) Parrot AR.Drone Device Research – research into the hardware, software, characteristics, support and community of the Parrot AR.Drone was conducted IOT understand these areas and their implications, and gain the ability to conduct the required procedures with the AR.Drone.
7) System Identification – procedures for system identification were researched IOT understand the intent of the process and choose an appropriate method for doing so.
8) Flight Tests – documentation on testing procedures and objectives were developed and flight tests conducted with a view to data collection for the system identification process.
9) Control Testing – control testing was again conducted, this time on the AR.Drone model, IOT to validate the model, validate possible controllers for the AR.Drone, and gain an understanding of the flight limits of the AR.Drone.

Figure 1. Parrot AR.Drone
II. PROJECT MANAGEMENT

Three systems engineering documents were produced during the first phase of the project including an SRD, FBL and ABL. These documents contained stakeholder requirements, the functional architecture and the preliminary physical architecture respectively, where ‘architecture’ refers to the framework in which, and baseline on which, this project was completed. In addition to the systems engineering documents guiding the technical aspects, two management and reporting tools were employed: the ITR management plan and the major undertaking reports.

The ITR management plan included methodology, test and evaluation and risk management documentation. Finally, reports were written for each major undertaking in the project such as the literature review, model derivation, controller testing and flight tests. These reports form the documentation behind the data pack – the project specific deliverable – and establish clear goals, and verifiable and repeatable results. The intention behind these reports is to answer likely questions of students continuing quadrotor research at the university and provide documentation for the MATLAB scripts and Simulink models created as a part of this project.

III. MATHEMATICAL MODEL

A mathematical model of a quadrotor was derived from first principles and implemented in Simulink. The purpose of this undertaking was to gain a conceptual understanding of the quadrotor system and to have the ability to test controllers in simulation. Much was gained from this undertaking, most importantly the realisation of the following:

1) The quadrotor is an under-actuated system in that it has six degrees of freedom (assuming rigid body dynamics [10]) but only four inputs. Lateral acceleration is an indirect result of non-zero attitude and cannot be controlled directly.
2) The four inputs can be defined in terms of sums and differences of rotor speeds. This allows the equations of motion to be written in four independent equations governing roll rate, pitch rate, yaw rate and vertical velocity.
3) The quadrotor is an inherently unstable system and requires adequate control to achieve useful motion. This inherent instability, however, also leads to greater manoeuvrability.

A quadrotor is a device which has four rotors arranged as in Figure 2. The forces and moments due to the rotation of the rotors directly control the angular acceleration and relative vertical acceleration of the quadrotor body and hence indirectly control lateral acceleration. Both the pitch and axis of rotation of the rotors is typically fixed.

A. Non-linear Model

A non-linear model was derived, linearised about the hover condition and then expressed in state-space form. The model was linearised so as to enable LTI system analysis and control. The linear model is sufficiently accurate since the quadrotor will not reach an attitude greater than 30° and hence not deviate greatly from the hover state. No literature exists that fully details the derivation of the model however as this project is to inform future research a fully documented derivation is included as a report in the project specific deliverable.

The derived equations below are similar to those presented by the GRASP Lab [4] and the China Jiliang University [11]. The Newton-Euler formulism was used, as opposed to the Lagrangian, due to the intuitive and recognisable equations \( (F = ma, \tau = I\alpha) \). The model does not consider higher order effects such as aerodynamic drag, blade flapping or gyroscopic torque such as in the paper by Huang et. al. [3] since an extremely high-fidelity model is not required.

Two frames of reference were defined: an inertial frame in which to write Newton’s equations and a non-inertial frame in which to write Euler’s equations. This was for convenience as Euler’s equations need not be written in an inertial frame and would be cumbersome and unintuitive if they were. The inertial frame was given the subscript \( W \), for world, or referred to by \( \{W\} \). Similarly, the subscript \( B \), and \( \{B\} \), were given to the non-inertial body-fixed frame whose origin is at the CoM of the quadrotor and whose axes rotate with the quadrotor.

The origin of \( \{B\} \) relative to \( \{W\} \) was described using the position vector \( r \). All these terms can be seen in Figure 2. Because the coordinates of the inertial frame are used all throughout the following mathematical equations, for clarity they are not given the subscript \( W \). Where coordinates are not in \( \{W\} \), this is made clear.

![Figure 2. Quadrotor Geometry, Frames of Reference, Forces and Moments [4]](image-url)
The rotational matrix \( R \) was defined to transform coordinates in \{B\} to coordinates in \{W\} using the Euler angles \( \phi \), \( \theta \) and \( \psi \) which refer to roll, pitch and yaw respectively. The angles are defined as follows: to get from \{W\} to \{B\} one rotates about \( z_W(=z_B) \) by \( \psi \), then about \( x_B(\neq x_W) \) unless \( \psi = 0 \) by \( \phi \), and finally about \( y_B \) by \( \theta \) [4]. Formally this is defined as the Z-X-Y Euler angles and the rotation matrix is

\[
R = \begin{bmatrix}
\cos \psi \cos \theta - \sin \phi \sin \psi \sin \theta & -\cos \phi \sin \psi & \cos \psi \sin \theta + \cos \theta \sin \phi \sin \psi \\
\cos \theta \sin \psi + \cos \psi \sin \phi \sin \theta & \cos \phi \cos \psi & \sin \psi \sin \theta - \cos \psi \cos \theta \sin \phi \\
-\cos \phi \sin \theta & \sin \phi & \cos \phi \cos \theta 
\end{bmatrix}
\]

**Equation 1. Rotational Matrix**

Newton’s equation, which holds only in the inertial frame, is below. Gravity acts in the \(-\hat{z}_W \) direction and the forces of the rotors act in the \( \hat{z}_B \) direction which is transformed into \( \hat{z}_W \) by the rotational matrix \( R \).

\[
m\ddot{r} = R \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \sum F_i \begin{bmatrix} \cos \psi \sin \theta + \cos \theta \sin \phi \sin \psi \\ \cos \phi \cos \psi \sin \theta - \cos \psi \cos \theta \sin \phi \\ \sin \phi \sin \theta - \cos \phi \cos \theta \sin \phi \end{bmatrix},
\]

**Equation 2. Translational Dynamics**

Equations of rotational motion are more conveniently formulated in \{B\} since the inertia tensor is time invariant and body symmetry can be used to simplify equations. Euler’s equation, \( \dot{\omega} = M - \omega \times \dot{\omega} \), was applied to the quadrotor and the rotational dynamics were derived. The equations are given in Equation 3. Since the origin of \{B\} is defined to be at the CoM of the quadrotor Sylvester’s law of inertia states that the products of inertia (those terms not on the diagonal) are zero. And as the quadrotor is approximately symmetrical about \( x \) and \( y \), \( I_{xx} \approx I_{yy} \). The inertia tensor is then a diagonal matrix and the equations can be simplified.

\[
I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L(F_2 - F_3) \\ L(F_3 - F_4) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} \dot{q} \\ \dot{r} \end{bmatrix} \times \begin{bmatrix} p \\ q \\ r \end{bmatrix},
\]

\[
I_{xx} \ddot{p} = Lk_x(\omega_2^2 - \omega_4^2) - qr(I_{xx} - I_{yy})
\]

\[
I_{yy} \ddot{q} = Lk_y(\omega_3^2 - \omega_1^2) - pr(I_{xx} - I_{zz})
\]

\[
I_{zz} \ddot{r} = k_m(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)
\]

**Equation 3. Rotational Dynamics**

**Equation 4. Simplified Rotational Dynamics**

Many institutions, including the GRASP Lab [4], use a simplified motor model where the upwards force and reactive torque is proportional to the square of the rotor speed and the response is governed by a first order differential equation. Subscript \( i \) differentiates between each motor/rotor.

\[
F_i = k_{F_i}\omega_i^2
\]

\[
M_i = k_{M_i}\omega_i^2
\]

\[
\dot{\omega}_i = k_m(\omega_i^{des} - \omega_i)
\]

**Equation 5. Motor Dynamics**

**B. Linearised Model**

The model was then linearised and expressed in state-space form. The rotor speeds are used to define inputs and hence will be used for control. The linearisation was carried out using the first-order Taylor series approximation. The operating points are

\[
\bar{\omega}_1 = \bar{\omega}_2 = \bar{\omega}_3 = \bar{\omega}_4 = \omega_h, \quad \omega_h = \frac{mg}{4k_p}
\]

**Equation 6. Hover State Operating Points**

The non-linear equations were linearised about the operating points and are given below.

\[
\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} g(\theta \cos \psi_0 + \phi \sin \psi_0) \\ g(\theta \sin \psi_0 - \phi \cos \psi_0) \\ 2k_F\omega_h(\omega_1 + \omega_2 + \omega_3 + \omega_4 - 4\omega_h) \end{bmatrix}
\]

\[
\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{2Lk_x\omega_h}{I_{xx}}(\omega_2 - \omega_4) \\ \frac{2Lk_x\omega_h}{I_{yy}}(\omega_3 - \omega_2) \\ \frac{2k_M\omega_h}{I_{zz}}(\omega_1 - \omega_2 + \omega_3 - \omega_4) \end{bmatrix}
\]

**Equation 7. Linearised Translational Dynamics**

**Equation 8. Linearised Rotational Dynamics**
C. State-Space Representation

State space representation is a form in which coupled first-order differential equations can be expressed. This form is useful since it is ubiquitous in system analysis techniques [12] and compatible with computer software such as MATLAB.

It was not possible to define the inputs as the rotor speeds since the constant term $4\omega_k$ in Equation 7 could not be included in the state-space representation. It was necessary to redefine the inputs in such a way that exploited the structure of the system. This is discussed below.

Four different motions can be performed independently by the quadrotor: vertical motion can be achieved by increasing the power of all rotors simultaneously, roll and pitch can be achieved by increasing and decreasing the speeds of opposing rotors and yaw can be achieved by causing a difference in speed and hence torque of rotors spinning opposite directions. These four motions occur due to sums and differences in rotor speeds and are completely decoupled from one another. This leads to the four inputs being defined as sums and differences of rotor speeds rather than the individual rotor speeds themselves.

The inputs are given the terms $\Delta \omega_F, \Delta \omega_\phi, \Delta \omega_\theta$ and $\Delta \omega_\psi$ and cause vertical motion, roll, pitch and yaw respectively. Their relationship to the individual rotor speeds is shown in Equations 9. The linearised equations were then written in terms of the four inputs $\Delta \omega_F, \Delta \omega_\phi, \Delta \omega_\theta$ and $\Delta \omega_\psi$ and are shown in Equation 10.

\[
\begin{align*}
\Delta \omega_F + \omega_h &= \frac{1}{4}(\omega_1 + \omega_2 + \omega_3 + \omega_4) \\
\Delta \omega_\phi &= \frac{1}{2}(\omega_2 - \omega_4) \\
\Delta \omega_\theta &= \frac{1}{2}(\omega_3 - \omega_1) \\
\Delta \omega_\psi &= \frac{1}{4}(\omega_1 - \omega_2 + \omega_3 - \omega_4)
\end{align*}
\]

Equation 9. Inputs and rotor speeds relationship

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z}
\end{bmatrix} = 
\begin{bmatrix}
ge(\theta \cos \psi_0 + \phi \sin \psi_0) \\
ge(\theta \sin \psi_0 - \phi \cos \psi_0) \\
\frac{2k_F \omega_h}{m} \Delta \omega_F
\end{bmatrix}
\]

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix} = 
\begin{bmatrix}
\frac{4Lk_F \omega_h}{I_{xx}} \Delta \omega_\phi \\
\frac{4Lk_F \omega_h}{I_{yy}} \Delta \omega_\theta \\
\frac{8\kappa m \omega_h}{I_{zz}} \Delta \omega_\psi
\end{bmatrix}
\]

Equation 10. Dynamics with redefined inputs

Notice that $\dot{z}, \dot{p}, \dot{q}$ and $\dot{r}$ are controlled directly by the inputs, and $\ddot{x}$ and $\ddot{y}$ are affected by the pitch and roll angles $\phi$ and $\theta$. For small $\phi$ and $\theta$, $\phi \approx p$ and $\theta \approx q$. Therefore, the inputs $\Delta \omega_\phi$ and $\Delta \omega_\theta$ control the snap of the quadrotor (fourth derivative of position).

D. Control Architecture and Controllers

The purpose of preliminary control tests on the ideal model was to validate the model, test the proposed control architecture and controllers, and investigate control theory issues such as instability due to delay in the feedback loops. The proposed architecture is quite common in literature [13] [14] and is shown in Figure 3.

PID controllers were selected for the following reasons:
1) They are used widely in literature [15] [11] [14] with positive results;
2) They are intuitive, simple to tune and robust to some noise and perturbations [16]; and
3) Higher order controllers are computationally more expensive, increasing the processing time of control commands.

The control architecture in Figure 3 was implemented in Simulink with PID and PD controllers for attitude and position control respectively.

PID controllers were not used for position control due to integral windup and overshoot issues. Integral control is usually implemented to remove steady-state error however since position is the second integral of acceleration which is proportional to the controlled variable, attitude, the steady state error will be zero and no anti-windup method is required.

The output of the position controller is limited so as to keep the quadrotor in near-hover conditions where the linear model is still sufficiently accurate. The maximum attitude of the AR.Drone, $30^\circ$, is used.

E. Attitude and Position Control

Attitude and position controllers were tuned and tested on the model. Results showed that with minimal manual tuning the quadrotor tracked static and dynamic positions with minimal error and in a manner equal to or better than human pilots. For example, when a forward movement of 10 metres was commanded the proportional gain initially dominated and pitched the quadrotor to $-30^\circ$. Between four and
六米处，控制器的偏导数部分开始起主导作用，并将四旋翼机倾斜。

这使得四旋翼机和其停止在10米处。

这是人类飞行员通常会直接控制四旋翼机的方式。

为了更好地可视化四旋翼机的运动，编写了一个MATLAB脚本，该脚本在3D中绘制四旋翼机（红色圆圈）、期望位置的演变（绿色线）以及当前期望位置（蓝色圆圈）。这些可以被视为一个MATLAB“电影”，其中四旋翼机的位置和方向都在显示。通过这种方式，可以可视化飞行剖面，如图4所示，其中四旋翼机正在跟踪动态位置。

F. Feedback Loop Delay

Nyquist plot was produced for varying amounts of delay in both the attitude and position feedback loops. It was found that the system was theoretical unstable with a delay of 105 ms however simulations showed that even at 70 ms of delay the quadrotor would flip over and hence was practically unsuitable. At a delay of 20 ms, which is what would be expected over a wireless link, the system was practically stable. The position control loop was much more resilient to delay and was practically stable even above 100 ms, well beyond anything that would be encountered in a real world system. In terms of the AR.Drone there is no issue with the attitude control loop as this processing is done on-board and delays would be much less than 20 ms.

IV. ASCENDING TECHNOLOGIES PELICAN QUADROTOR

Initially it was the intention of this project to model and control the AscTec Pelican. After beginning work with the Pelican it was discovered that the documentation was not appropriate for guiding an undergraduate student and that part of the XBee wireless network system had not shipped with the device and so it was not possible to remotely capture data from the Pelican.

Such problems associated with the Pelican had already been identified as a risk in the ITR management plan with the solution being to instead continue the same work using the AR.Drone platform.

While around three weeks were spent working with the Pelican without significant results, much was learned during the initial familiarisation such as working with the on-board Ubuntu operating system and charging procedures for Li-Po batteries. This proved valuable in the understanding of the AR.Drone system.

V. PARROT AR.DRONE QUADROTOR

A. Overview

The Parrot AR.Drone is a COTS quadrotor system released in 2010 marketed as part of an augmented reality video game. The device retail for between AUD$200-$300 [17]. Parrot has released software [18] for both iOS and Android based mobile devices that allow users to connect to the quadrotor over WiFi and play games such as battling other drones, with the ability to shoot at another drone within the game (the actual device does not shoot).

A software development kit [19] is freely available allowing development of further iOS and Android apps as well as Linux-based applications. In this way it is also possible to control the AR.Drone from a PC running Linux using input methods such as joysticks or handheld controllers.

B. Control Inputs

The device abstracts much of the complexity of control. Users are able to press buttons to command actions such as ‘take off’ and ‘land’, where the onboard closed loop control system performs all necessary actions. The ‘take off’ command, for example, performs the following:

1) Starts all motors
2) Increases thrust of all rotors to increase altitude until stable at around 1 m
3) Adjusts rotor speeds to maintain zero attitude and zero yaw
4) Monitors downwards facing camera to keep drone above take off point
The user is also able to send the following commands; and when there is no user input the AR.Drone automatically returns to and maintains the hover state.

1) Pitch Angle (about lateral axis) (max 30 deg)
2) Roll Angle (about longitudinal axis) (max 30 deg)
3) Yaw Angle Rate (about vertical axis) (max 100 deg/s)
4) Vertical Velocity (in body fixed frame) (max 1 m/s)

C. Hardware and Software

As stated by Bristaeu et. al. [20], the AR.Drone ‘is a prime example of sophisticated use of low-cost sensors (MEMS and cameras) for mass markets where retail price is of major importance’. For its relatively low cost the AR.Drone has impressive hardware and control capabilities. Only four types of sensors are used:

1) A 3-axis MEMS accelerometer is used to measure lateral accelerations;
2) A 1 and 2-axis gyro are used to measure angular accelerations. The 1-axis gyro is used to measure yaw-rate and is of a higher quality so as to minimise yaw-drift during flight;
3) 2 ultrasonic sensors are used to measure altitudes out to 6m; and
4) A downward facing camera is used for velocity estimation using optical flow algorithms.

A 40 MHz PIC digitises sensor inputs and a 468 MHz ARM9-core processor running a Linux-based ROS manages Wi-Fi communications, video data sampling, video compression (for wireless transmission), image processing, sensors acquisition, state estimation and closed-loop control. The data acquisition and control thread is sequenced by the PIC ADC and runs at 200 Hz [20]. This data can be accessed in real time over WiFi from the device and used by off-board controllers. Also integrated onboard is a hardware DCT to enable faster processing of video and navigation data.

Perhaps the most impressive feat by Parrot is the implementation of the vision algorithms for lateral velocity estimation. The video feed from the downward facing camera is processed on-board using two complementary techniques: full-frame multi-resolution optical flow and corner tracking. The first method is best suited to low-contrast environments, has a constant computational cost and is used by default. The second method gives increased accuracy and is automatically employed at low speed and when sufficient information is contained in the image to track above a threshold number of objects [20]. The AR.Drone is excellent at maintaining its lateral position due to this vision based control however adequate ground lighting is required for proper operation.

D. The AR.Drone as a Research Platform

The AR.Drone is particularly suited to being a platform for research by the UAV Laboratory for the following reasons:

1) The AR.Drone is a COTS system and as such a saving is made in both time and money. The design risk is assumed by the manufacturer, Parrot, the device comes with a warranty, is readily available on the commercial market (no development time) and support is available from the manufacturer.
2) Parrot is actively supporting community development and has made available an SDK for off-board applications and the source code for the onboard operating system. This negates the need to ‘hack’ the platform and has resulted in an active online community of university reserachers and backyard hobbyists experimenting with the AR.Drone.
3) The complexity of control is abstracted leaving only four intuitive inputs (pitch, roll, yaw rate and vertical velocity). As such, researchers are able to concentrate on their particular area of research without worrying about problems such as sensor fusion and optical flow algorithms.

VI. System Identification

‘System identification is the art and science of building mathematical models of dynamic systems from observed input-output data. It can be seen as the interface between the real world of applications and the mathematical world of control theory and model abstractions’ [21].

When controlling the AR.Drone off-board, the system is essentially a ‘black box’ as visualised in Figure 5. Control inputs are sent from the client device, the device responds in movement, this movement is captured as navigation data by the on-board sensors and the navigation data can be read by the client device.

By treating the AR.Drone as a black box and characterising the inputs and outputs, complex dynamics are inherently modelled. For example, the response time of the motors, blade flapping, gyroscopic torque of the rotors, body inertia, body drag and the on-board closed loop control system are all inherently modelled when characterising the inputs and outputs of the AR.Drone. This approach to modelling the AR.Drone is identical to the common procedure in electrical engineering of characterising systems by measuring parameters such as input impedance, output impedance, power usage, S-parameters and SNR without
regard of the specific internals of the system. The model of the AR.Drone can then be treated just as an undergraduate student treats the model of the operational amplifier, and higher level work can commence with the AR.Drone in areas such as vision-based autonomous navigation.

Within the flight limits of the AR.Drone the input-output relationships can be approximated as independent LTI systems. This approximation was assumed to be valid due to structure of the AR.Drone system and was validated when flight data was recorded as discussed in subsection C.

In characterising the system a number of approaches are valid, such as frequency domain analysis using sinusoids or chirps, or time domain analysis such as step responses. Due to the available inputs it was determined that the most appropriate technique would be a step response analysis.

A. Software Development Kit

The Parrot AR.Drone SDK 2.0 is freely available on the internet [19] and requires a Linux-based operating system. Ubuntu 12.04 was installed on a laptop and the SDK was downloaded and compiled.

The SDK contains a number of libraries to enable applications to be written and have functionality such as live video processing, joystick and handheld controller input processing and navigation data acquisition and display.

Developing software using the SDK was determined not to be feasible in the time constraints of the project, particularly due to the large overhead in familiarisation with the programming language and the libraries provided by Parrot. Fortunately a number of demonstration programs were included, one of which enabled both control by handheld controller and automatic recording of navigation data. This functionality was essential in the system identification process. A compatible joystick was not available however by modifying the SDK joystick XML file, PS3 controller buttons were able to be mapped to AR.Drone controls. With a wireless card, Ubuntu, the Parrot SDK and a PS3 controller it was possible to control the AR.Drone and have all navigation data recorded to a text file at a sampling rate of 200 Hz. It was then possible to commence flight tests and the system identification process.

B. Flight Tests

A suitable location was required in which to conduct flight tests. Adams Hall on the ADFA precinct was chosen for the following reasons:

1) The indoor location ensured that wind did not affect results;
2) The volume of the useable indoor area (approximately 20 m × 20 m × 10 m) enabled sufficiently long flights such that lateral and vertical velocities reached steady state;
3) No objects protruded above the level floor which could cause erroneous altitude measurements; and
4) The distinct pattern of the floor, when lit, ensured velocity measurements based on the on-board optical flow algorithms were as accurate as possible.

Flight tests consisted of applying step inputs to the four control inputs of the AR.Drone. Step inputs were only applied when the AR.Drone was in a stable hover state and in a position and altitude such that its resultant motion would not be affected by ground effect or wash from adjacent walls. A requirement of the test sequences was that recorded navigation data was representative of the typical behaviour of the AR.Drone.

C. Primary Effects of Control

The primary effects of control refer to the effects which are the intended result of applying a control input. For example, when commanding a desired pitch angle it is intended that the AR.Drone change attitude to assume that pitch angle, and as a result accelerate longitudinally (forwards/backwards) in the direction of negative pitch – proportional to \( \sin(\theta) \)

where \( \theta \) is the angle of pitch. As such the primary effects of a pitch input could be considered to be pitch and longitudinal acceleration. While this is strictly the case, it was not appropriate when modelling the AR.Drone.

The ideal mathematical model did not consider drag and so when the quadrotor was pitched, the quadrotor body accelerated ad infinitum and so the indirect result of pitch could be considered as being a steady state longitudinal acceleration. This was not

![Figure 6. Pitch Response Model Validation](Image)
seen with the AR.Drone however since the drag on the AR.Drone body resulted in a steady state velocity when the longitudinal component of thrust and drag forces were equal. As stated in the section introduction, with reference to Figure 5, the intention of the system identification was to model input-output characteristics and so inherently model effects such as body drag. In this regard it is clear that since the longitudinal velocity reaches a steady state given a fixed pitch angle (it produces an expected step response), it is appropriate that longitudinal velocity be considered as the indirect primary effect of a desired pitch angle. Similarly, the indirect primary effect of a roll angle, \( \phi \), is a lateral velocity. Table 1 details the primary effects of the four AR.Drone inputs.

<table>
<thead>
<tr>
<th>Control Input</th>
<th>Direct Primary Effects</th>
<th>Indirect Primary Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch</td>
<td>Pitch</td>
<td>Longitudinal Velocity</td>
</tr>
<tr>
<td>Roll</td>
<td>Roll</td>
<td>Lateral Velocity</td>
</tr>
<tr>
<td>Yaw Rate</td>
<td>Yaw Rate</td>
<td></td>
</tr>
<tr>
<td>Vertical Velocity</td>
<td>Vertical Velocity</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Primary Effects of Control

The navigation data was imported into MATLAB and processed using the system identification tool. In determining the transfer function between the inputs and outputs, the following standards and procedures were followed:

1) Each transfer function was calculated in a separate MATLAB script file;
2) All post processing done by the script was performed in the workspace without affecting the original data. This ensures that the script can be run again and produce identical results;
3) During flight tests, identical manoeuvres were conducted multiple times since in some cases individual recorded movements were not truly representative of the AR.Drone’s flight. Data from multiple identical manoeuvres could be combined to form a step response truly representative of the actual response. In most cases erroneous results were due to sensor dropout and by considering only the envelope of multiple identical manoeuvres, quality results were achieved;
4) All data was stored in an *iddata* data structure which enables easy processing by MATLAB's array of LTI system tools, most importantly *ident*, the system identification tool.
5) The number of poles a given transfer function should have was informed by the mathematical analysis of a quadrotor system done in the initial phases of the project.
6) The response of the AR.Drone was compared to the response of the fitted transfer function for the same step input (see Figure 6).
7) The transfer function response was judged to be representative of the AR.Drone if and only if the transient response, the overshoot and the steady state response were all representative of the recorded navigation data from the AR.Drone. As an example, all three of these characteristics can be seen to be validated in Figure 6 which shows the relationship between the desired pitch command and the actual pitch of the AR.Drone.

D. **Secondary Effects of Control**

‘Secondary Effect’ is a term used in General Aviation to describe the motion of an aircraft that is the result of a primary effect, and not the intended effect of the respective control input. For example when considering a fixed wing aircraft, the secondary effect of yaw is roll. As the aircraft yaws the outside wing, travelling faster than the inside wing, produces more lift and hence causes a roll in the direction of yaw.

Helicopters also exhibit secondary effects such as roll due to forward velocity. The advancing blade produces more lift than the retreating blade during forward flight and hence creates a moment about the longitudinal axis causing roll.

The secondary effect exhibited by the quadrotor is illustrated in the plot of recorded navigation data in Figure 8. A pitch command is sent at approximately 14.1 s and the pitch is the first to respond, reaching a steady state in approximately 0.5 s. The forward velocity increases as a result, although relatively slowly, reaching a steady state after approximately 1 s.

![Linear Simulation Results](image)

Figure 7. Yaw Rate Response Model Validation

8

Final Project Report 2012, SEIT, UNSW Canberra
During this manoeuvre the AR.Drone loses altitude. This occurs as a result of two factors: firstly, the vertical component of thrust is reduced when the AR.Drone is at a non-zero attitude, and secondly, the incident wind adds to the downward force on the AR.Drone body during forwards flight. This secondary effect is not compensated for by the AR.Drone and hence must be modelled.

The downwards velocity reaches a steady non-zero value during forward flight and hence negative vertical velocity was considered to be the secondary effect of both roll and pitch control inputs and was included in the Simulink model of the AR.Drone.

It should be noted that other secondary effects such as roll due to forward velocity were not observed. In the case of roll due to forward velocity, this is most likely not observed due to the AR.Drone actively maintaining a zero roll angle during forward flight as commanded by its onboard closed loop control system.

E. Other Characteristics

Two interesting characteristics of the AR.Drone were observed in addition to the primary and secondary effects of control and were related to the function of the downward facing camera.

1) In a stable hover the AR.Drone monitors the optical flow/tracker objects using its downwards facing camera and aims to maintain zero velocity. It also uses this information to ensure accelerometer and gyro drift is minimised over time and it is able to determine its attitude. In one flight test the lights in the test area were turned off and the AR.Drone was flown to an altitude at which it was not able to determine ground patterns, not able to accurately determine its velocity and not able to zero the drift of the accelerometers and gyros (around 10 m). Within approximately 3 s the AR.Drone became unstable in its hover and began oscillating violently in pitch and roll. Within another 2 s the AR.Drone had crashed to the ground. The result of this test showed that the downward facing camera is pivotal in the stable operation of the AR.Drone and that the sensor data collected from the INS suffers from both noise and drift, and must be heavily filtered by fusing measurements with data from the downward facing camera.

2) In a similar scenario a human established themselves under the AR.Drone presenting a high-contrast target to track. The AR.Drone maintained a stable hover position. If the human moved the AR.Drone interpreted this as a relative movement by itself and rapidly corrected. In this way the AR.Drone would follow the human around the room, maintaining a stable hover above them. Interestingly, the AR.Drone would violate its own attitude limits in correcting for perceived movement due to movement by the human. For example, if a pitch/roll limit of 12° was set (this parameter is sent to the AR.Drone and stored on-board), the AR.Drone would still pitch/roll to approximately 30° during its attempt to stay above the moving human.

F. Controller Testing

The AR.Drone model was implemented in Simulink with a PD position controller. Control tests were run and it was found that the AR.Drone model was able to track static and dynamic positions with minimal error.

One limitation of the AR.Drone, however, is the amount of thrust able to be produced by the rotors. From a hover position the drone is able to ascend at a rate of approximately 1 m/s, however when at a non-zero attitude this rate is reduced, to a point where at 30° of either pitch or roll the AR.Drone cannot attain a positive vertical velocity and can only simply maintain a constant altitude.

VII. Conclusion

During the conduct of this project a number of important results and outcomes were obtained. Perhaps most importantly a validated model of the Parrot AR.Drone was developed and implemented in Simulink. Leading to this result, however, a full derivation of the mathematical quadrotor model was derived, a control architecture and PID/PD controllers were tested and validated, and research and analysis was conducted into the use of the Parrot AR.Drone as a research platform. It was found that the Parrot AR.Drone is a capable system; it has adequate support from both Parrot and the online community and is suitable for use as a research platform. In achieving these outcomes this project has achieved its overall aim of both establishing a body of knowledge on the subject of quadrotors and deriving and developing a validated model of the Parrot AR.Drone.
VIII. RECOMMENDATIONS

The Parrot AR.Drone has recently been superseded by the Parrot AR.Drone 2.0 which is available for less than AUD$300 [22]. The AR.Drone 2.0 has updated sensors including a 720p forward facing camera. All the techniques used for the system identification done in this project are directly applicable to the AR.Drone 2.0 and so can be conducted very easily using the major undertaking reports as a guide. Since the aim of the UAV Laboratory is to conduct vision-based research the advantages of having a higher definition camera are palpable. The scope of this project did not include software development using the Parrot AR.Drone SDK 2.0 due to the overhead in familiarisation with the programming language and AR.Drone libraries. The next logical step in control of the AR.Drone 1.0/2.0 is to develop a program to allow both manual and automatic control. It should be noted that software built using the Parrot AR.Drone SDK 2.0 is compatible with both versions of the AR.Drone – the numbering of the SDK version is independent.

Recommendations:

1) Purchase and model the Parrot AR.Drone 2.0. Additional batteries should be purchased so as to enable longer flight times when capturing navigation data. Flight times vary however batteries last for approximately 15 minutes.

2) Develop or adapt software for manual and automatic control of the AR.Drone 2.0 using the Parrot AR.Drone SDK 2.0.

Both of these activities are suitable as a single fourth-year electrical engineering thesis in 2013.

IX. ACKNOWLEDGEMENTS

I wish to thank my supervisors, A/Prof Valeri Ougrinovski and Dr Matthew Garratt, my family and friends for their support throughout the year. I also wish to thank the School and Engineers Australia for inviting me to present a seminar on my thesis at the 2012 ITEESPAN Awards Night.
X. REFERENCES


11
Final Project Report 2012, SEIT, UNSW Canberra


