INVESTIGATIONS IN THE CONTROL OF A FOUR-ROTOR AERIAL ROBOT

ABSTRACT

This paper presents the challenges and successes of developing several control architectures on a custom designed quadcopter. In order to achieve this goal, it was necessary to use parameter identification to create a model of the kinematic equations, simulate the model with several control architectures in MATLAB/Simulink and finally test the algorithm on the actual hardware. The results of the simulation indicate that it is possible to design a controller to allow the quadcopter to hover stably. However, in order to implement this architecture on the hardware and control the attitude, it is necessary to have more information about the sensor data, more accurate parameter identification of the model and more robust control architecture to handle uncertainty in the plant.

KEY WORDS

robot, quadcopter, control architectures, LQR, PD control

quadrotor, parameter identification

1. Introduction

Unmanned aerial vehicles (UAV) have garnered increased interest in the last decade due to their utility in military and commercial applications. These include surveillance, search and rescue, remote inspection or any application that would be dull, dirty or dangerous [1, 2].

The UAV used in this study falls in the category of heavier than air motor vehicle with vertical takeoff and landing (VTOL) [1]. The advantage of this type of configuration is that it can function well in densely populated environments with close proximity to debris, humans or other robots [3]. The primary benefits to this type of VTOL UAV are the excellent payload to power ratio, vertical, stationary and low speed flight and the ability to maneuver in narrow spaces [3].

Another name for a four rotor aerial vehicle is a quadcopter. The quadcopter provides an ideal option over other VTOL vehicles such as helicopters because of the increased thrust, maneuverability, safety, stability and low cost implementation [4, 5]. Unlike a helicopter, a quadcopter does not require a tail rotor to compensate for a yaw motion and thus all energy contributes to the upward thrust with reduced mechanical vibration. This design also makes quadcopters safer and they can be in closer proximity to humans. Lastly, the control of a quadcopter is more intuitive for a remote human operator [6].

However, a quadcopter is still an under-actuated nonholonomic vehicle since there are only four parallel force inputs to control its six output coordinates (position and orientation in space) which requires complex control algorithms.

2. Background

Quadcopters themselves are not novel but techniques for controlling them are [1-6]. Typically, the position is controlled by a remote operator and the attitude is automatically stabilized by an onboard controller. Some of the control approaches presented in the literature include output tracking, state parameter, feedback linearization, neural networks, nonlinear as well as hybrid control schemes for attitude stabilization and torque compensation. In [6], they were able to prove that a PD controller can provide asymptotic stability which was examined in this work. This paper will provide a guide on how to use scientific methods such as parameter identification and kinematic modeling to design, develop, simulate and test control architectures on a quadcopter.

2.1 Hardware Design

The quadcopter used for this study was designed by a group of undergraduate students for their senior design project. Their ultimate goal was to create an autonomous flying robot to take samples and measure air quality at various altitudes in the atmosphere. Although they were able to design and build the quadcopter, they were not able to create a controller in order for the robot to stably hover and eventually take flight. This is where this work begins.

The quadcopter frame was designed to minimize weight and maximize structural integrity and to support a desired payload. The dimensions of the quadcopter were less than 0.5m x 0.5m. It was made of lightweight aluminum with a central mounting board made of lightweight, nonconductive acrylonitrile butadiene styrene (ABS) to insure electrical isolation, account for noise and vibration, and allow for isolated mounting of essential flight sensors. Figure 1 provides a graphic of the quadcopter's mechanical design.

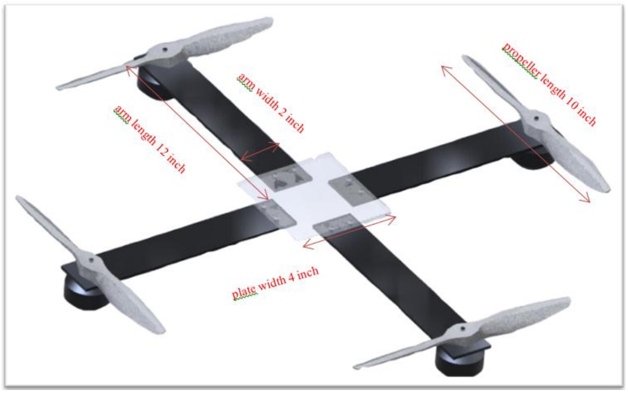

Figure 1. Quadcopter's mechanical design

The quadcopter platform is equipped with four brushless DC motors, each of which required an electronic speed controller (ESC). They were chosen because of their high torque output to minimal power input ratio and light weight compared to other models. The propeller blade sizes were chosen to be 10 x 7 inch (254 x 152 mm) in order to meet the necessary thrust requirement.

2.2 Electrical Design

The quadcopter was required to hover and remain stable at various heights, it had to handle external disturbances such as wind gusts as well as being able to navigate to a predetermined target. Therefore, the electronics on board the quadcopter consisted of two major systems: inertial navigation and external sensors. The electronics board also had a wireless communication system for data transmission and to communicate with the robot from a ground station.

The inertial navigation system (INS) consists of accelerometers and gyroscopes to detect position, orientation, and velocity. Furthermore, the quadcopter also has an altimeter, compass and GPS to enable precise outdoor navigation. The primary components used for this study were the INS sensors. However, the accuracy of the INS is limited by the sampling frequency and suffers from external vibration. For gyroscopes, it is important to have sufficient sensing range with enough precision to avoid sensor clipping when performing quick maneuvers.

In order to interface with the quadcopter, a four layer control board was developed with a 1.8V power plane, a 3.3V digital plane, a 3.3V analog plane, and a separate ground plane. This board could run embedded C code at 150 MHZ and connect to a ground station via TCP/IP over a standard 802.11g network. The wireless link has an effective range of 75 meters. It outputs PWM signals on four channels 50 times a second and reads its six 12-bit analog bits with a 12.5 MHZ sampling rate.

A Texas Instruments "TMS320F28335 Delfino Floating-Point Digital Signal Controller" 32-bit 150 MHz processor was used. The 28335 is a C2000-class real-time digital signal controller that includes many hardware-accelerated math functions, several internal timers and ADC and UARTS. The processor provided a MATLAB/Simulink interface to fully support the development of high level control architectures. Mathworks "Embedded Coder" toolbox and TI's "Code Composer Studio v.4" were used to implement embedded C code for this work.

Ground bounce, voltage ripples, or operation under heavy load, could drop the primary flight battery below the required 3.3V supply voltage for the onboard regulator. To avoid a control board brown-out, a smaller secondary battery was attached to the robot to power the avionic, control, and communication circuits.

3. Controller Design

This section will detail the necessary steps to develop a controller for the quadcopter described in section 2.

3.1 Dynamic Model and Parameter Identification

Due to the complexity of the quadcopter and the fact that it is an under-actuated aircraft with fewer controllable degrees of freedom than possible degrees of freedom, it is necessary to make simplifications and abstractions to derive the kinematic model. The simplifications were that the aircraft was modeled as a symmetric, rigid cross frame structure equipped with four rotors in a common horizontal plane. Assume that the robot has a local reference frame, $\mathcal{J}$, coincident with the center of mass of the aircraft and an inertial reference frame, ξ. The two reference frames are related to each other by a rotation matrix dependent on Euler angles, ϕ, θ, ψ, to express the robot's orientation in space.

This work will use the direction cosine matrix (DCM) approach to represent the aircraft's orientation with respect to the inertial reference frame [7]. The DCM fits more naturally than the quaternion approach for control and navigation purposes. It is more intuitive due to the mathematical familiarity of coordination transformation of a vector in one system to another. The following DCM transforms the quadcopter's three axis vectors to be coincident with the inertial reference frame. Note that $c\,\theta$ denotes $\cos\theta$ and $s\,\theta$ denotes $\sin\theta$, etc.

$$\begin{bmatrix} (c\theta * c\psi) & (c\psi * s\theta * s\phi - s\psi * c\phi) & (c\psi * s\theta * c\phi + s\psi * s\phi) \\ (c\theta * s\psi) & (s\psi * s\theta * s\phi + c\psi * c\phi) & (s\psi * s\theta * c\phi - c\psi * s\phi) \\ (-s\theta) & (s\phi * c\theta) & (c\phi * c\theta) \end{bmatrix}$$

The main effects modeled were the aerodynamics (propeller rotation), inertial counter torques (due to changes in propeller rotation speed), gravity effects (from the center of mass of the robot), and gyroscopic effects.

To understand the kinematics, note that $V = [\dot{x}\ \dot{y}\ \dot{z}]^T$ stands for the aircrafts linear velocity and $\Omega = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T$ for the angular velocity, both expressed in the local reference frame. Newton's equations of motion yield the following equations of motion for the quadcopter:

$$\dot{\xi} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R * V \tag{3.1}$$

$$\dot{V} = \ddot{\xi} = -\Omega \times V + g * R^T * \overrightarrow{e_3} - \frac{1}{m} * T * \overrightarrow{e_3} \tag{3.2}$$

$$\dot{\zeta} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \Omega \tag{3.3}$$

$$I * \dot{\Omega} = -S(\Omega) * I * \Omega - \mathcal{G}_a + \tau_a \tag{3.4}$$

Here $S(\Omega)$ denotes a skew symmetric matrix which helps calculating vector cross products.

$$S(\Omega) = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \tag{3.5}$$

and

$$\dot{R} = R * S(\Omega). \tag{3.6}$$

$\mathcal{G}_a$ denotes the gyroscopic torques and τ_a the external airframe torques.

Let $T = \sum_{i=1}^{4} |f_i|$ be the total thrust with each propeller lift force, $f_i = -b * {\omega_i}^2$. Thrust and drag of the propellers result in external airframe torques.

$$\tau_a = \begin{bmatrix} l * b * ({\omega_4}^2 - {\omega_2}^2) \\ l * b * ({\omega_3}^2 - {\omega_1}^2) \\ \kappa * ({\omega_2}^2 + {\omega_4}^2 - {\omega_1}^2 - {\omega_3}^2) \end{bmatrix} \tag{3.7}$$

with l as the distance to the center of gravity, and κ and b are two parameters mainly depending on the density of air as well as the size and shape of the propellers.

Gyroscopic torques account for the angular moments of spinning masses. The gyroscopic torque,

$$\mathcal{G}_a = \sum_{i=1}^{4} J_r * (\Omega \times \overrightarrow{e_z}) * (-1)^{i+1} * \omega_i \tag{3.8}$$

with J_r as the inertia for each rotor.

The above equations had to be simplified and consolidated to derive a system of differential equations that describe the motion of a quadcopter. Next, the DCM was used to simplify the equations. The derivative of equation (3.1) was given by

$$\ddot{\xi} = R * \dot{V} + \dot{R} * VI * \dot{\Omega} = -S(\Omega) * I * \Omega - \mathcal{G}_a + \tau_a \tag{3.9}$$

The substitution of equations (3.2) and (3.6) into equation (3.9) yields

$$\ddot{\xi} = -R * \Omega \times V + R * R^T * g * \overrightarrow{e_3} - \frac{1}{m} * R * T * \overrightarrow{e_3} + R * S(\Omega) \tag{3.10}$$

which due to the nature of R and the skew-symmetric matrix simplifies to

$$\ddot{\xi} = g * \overrightarrow{e_3} - \frac{1}{m} * R * T * \overrightarrow{e_3}. \tag{3.11}$$

Finally, this derivation provides the 2nd order differential equations for the aircraft's position and orientation in space.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} -\frac{b}{m} * ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) * (c\psi * s\theta * c\phi + s\psi * s\phi) \\ -\frac{b}{m} * ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) * (s\psi * s\theta * c\phi - c\psi * s\phi) \\ g - \frac{b}{m} * ({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) * (c\phi * c\theta) \end{bmatrix}. \tag{3.12}$$

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\dot{\theta} * \dot{\psi} * \left(\frac{I_{zz} - I_{yy}}{I_{xx}}\right) - \frac{J_r}{I_{xx}} * \dot{\theta} * (\omega_2 + \omega_4 - \omega_1 - \omega_3) + \frac{l*b}{I_{xx}} * ({\omega_4}^2 - {\omega_2}^2) \\ -\dot{\phi} * \dot{\psi} * \left(\frac{I_{xx} - I_{zz}}{I_{yy}}\right) + \frac{J_r}{I_{yy}} * \dot{\phi} * (\omega_2 + \omega_4 - \omega_1 - \omega_3) + \frac{l*b}{I_{yy}} * ({\omega_3}^2 - {\omega_1}^2) \\ \dot{\phi} * \dot{\theta} * \left(\frac{I_{yy} - I_{xx}}{I_{zz}}\right) + \frac{\kappa}{I_{zz}} * ({\omega_2}^2 + {\omega_4}^2 - {\omega_1}^2 - {\omega_3}^2) \end{bmatrix} \tag{3.13}$$

Before any adequate controller could be developed, the physical parameters of the quadcopter had to be measured or at least roughly estimated. It was divided into the following categories: mass and dimensions of the aircraft, dynamics of rotors, motors and airframe and aerodynamics of the propellers.

For a thorough analysis of motor dynamics on the quadcopter, it is necessary to derive a mathematical model based upon well-known equations of motion for DC motors. However, since there was no information available regarding the motor constants and it is rather complicated to model brushless DC motors, an alternate approach was used. This approach involved deriving the frequency-to-speed relationship for each motor.

The experimental setup included one motor equipped with a reflective piece of tape and affixed to the ground. Then a voltage generator using PWM input signals was attached to the motor. The input signal was then incrementally increased and the output speed (RPM) signal was measured using an infrared non-contact tachometer.

In theory, the frequency-to-speed relationships of BLDC motors is linear but speed controllers need a way to determine the rotor position to direct the rotation. This is done by measuring the back EMF in the undriven coils of the motor. If the difference in back EMF between two input values is not high enough the controller doesn't change the RPM which results in a staircase function. Figure 2 illustrates this phenomenon.

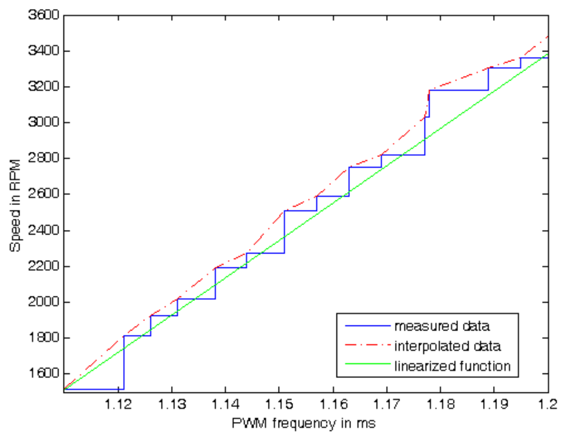

Figure 2. Frequency-to-speed relation of brushless motors

Since the dynamics of the quadcopter were highly dependent on its moment of inertia, a pendulum experiment was used to determine the moments of inertia for the entire robot. Since the center of mass and the principal axes (coordinate axes) of the aircraft are known, the moment of inertia about each principal axis was found directly by developing a torsional pendulum with the rotation of the pendulum passing through the center of mass aligned with the axis of rotation. Therefore the mass moment of inertia can be calculated using the equation of a simple harmonic oscillator (linearized around its equilibrium point $\theta = 0$).

$$I = \frac{m*g*(l+d)*t^2}{(2*\pi)^2} - m*(l+d)^2, \qquad (3.14)$$

where t is the period of oscillation in sec. The experimental setup is shown in figure 3.

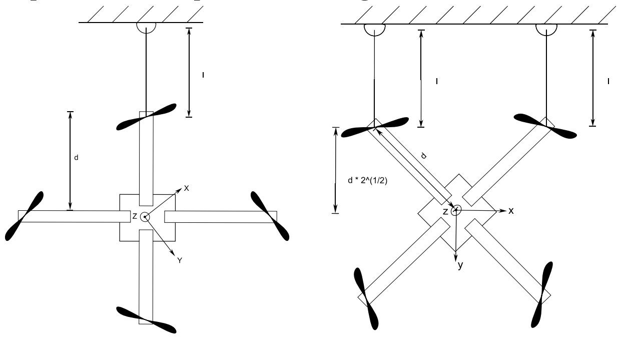

Figure 3. Experiment to measure moments of inertia

For the moment of inertia of each rotor, geometry and basic mechanical equations were used to calculate both values. It was composed of two parameters, the rotational moment of inertia around the propeller axis, $J_{propeller}$ and the rotational moment of inertia around the motor axis, J_{motor}. The motor was modeled as a solid cylinder. Since the gyroscopic torques only affect the dynamics around the aircraft's z-axis, only J_z was required. To estimate the moment of inertia for a propeller blade, its mass and geometry were required. Because all propeller blades are symmetric, only one had to be measured. Again only J_z was required. Each propeller blade was modeled as a thin rectangular plate with equal mass distribution.

The two most important aerodynamic coefficients are drag factor, b and thrust factor, κ. Pounds et al. [5] derived a relationship between thrust and induced velocity in a rotor (see equation (3.16)). They also derived a relationship between drag and induced velocity in a rotor (see equation (3.17)). In a stable hovering position, the aircraft's total thrust T can be calculated with

$$T_{hover} = \sum_{i=1}^{4} 2 * \rho_{air} * A * {v_{i_{hover}}}^2, \qquad (3.15)$$

where ρ_{air} denotes density of air, A is the disc area of a rotation propeller blade and v the induced linear velocity. The linear velocity at each point along the propeller blade is proportional to the radial distance from the rotor shaft. Thus, by integrating along the length of the blade, rules for the entire rotor using and the rotor's angular velocity

ω can be produced which show a proportional relationship.

$$T_{hover} = C_{thrust} * \rho_{air} * A * (\omega * r)^2 \tag{3.16}$$

$$Q_{rotor} = C_{drag} * \rho_{air} * A * (\omega * r)^2 * r \tag{3.17}$$

Here r denotes the radius of each rotor. Experiments were designed to measure the relationship between thrust (respectively drag) and angular velocity, ω.

One can see that thrust is proportional to the square of the angular velocity. The total thrust of the quadcopter in a stable hover flight has to equal the total mass, m of the robot times the acceleration of gravity, g. It was measured using an infrared non-contact tachometer and then the thrust coefficient was calculated.

To estimate the drag factor, κ, it was necessary to describe the torque acting on the shaft of each motor in relation to the angular velocity, ω. In an experimental setup, a single rotor was fixed on a plate and attached to the ground. It was supplied with an 11V battery voltage. To estimate the drag coefficient, the induced current at hover speed with and without a load (propeller blade) was measured. By measuring the overall power loss, the induced torque caused by the attached propeller blade can easily be calculated with

$$\tau_{loss} = \frac{P_{loss}}{\omega_{hover}} \tag{3.18}$$

To get the total drag, all four rotors had to be considered. As for thrust, the drag is proportional to the square of the angular speed, ω, of the rotors. This results in a proportional drag coefficient, κ of

$$\kappa = \frac{4 * \tau_{loss}}{{\omega_{hover}}^2} \tag{3.19}$$

The control of an UAV system often contains two main control loops: an inner, faster, vehicle control loop, and an outer, slower, mission control loop. The mission control loop calculates the desired position and velocity for the vehicle, which is then stabilized by the vehicle control loop. As in most mobile robotics applications, direct position control is often not feasible due to position measurement (or estimation) not being accurate enough for a feedback controller [8].

The focus of this research was on the attitude dynamics (the inner control loop) since it is necessary to stabilize the aircraft in flight and hover at a current position before any position control can be designed and tested.

To establish a vehicle controller, the derived state space model was decomposed into two subsets of differential equations to describe the dynamics of the attitude angles and the translation of the UAV. Examining equations (3.12 - 3.13) one can see that the dynamics of the system angles as well as their derivatives do not depend on translational components whereas the translational movement depends on the angles but not on the angular movements. Therefore the overall system can be broken into two subsystems: an angular rotation subsystem and a translational movement subsystem.

The angular rotations subsystem consists of the last six components (equation (3.13)); roll, pitch, and yaw angles and their derivatives. There are two common control approaches often used in current research on the control of quadcopters, PD and optimal or LQ control (or slight variations of these principles). Bouabdallah et al. used a Lyapunov equation to prove that a PD controller that could stably hover a quadcopter at any desired height z_{hover} can be developed for an initial condition where $cos(x_7) * cos(x_9)$ do not equal zero [1].

Both control architectures presented here used a linearized state space model of the quadcopter. This method is sufficient most of the time since a quadcopter is primarily used in hover flight mode. However, strong winds or aggressive maneuvers could lead to a loss of control.

3.2 State space PD feedback controller

The first step in the state-space design method for a PD controller was to find the law as a feedback of a linear combination of the state variables

$$u = -K * x = -\begin{bmatrix} k_{1,1} & \cdots & k_{1,6} \\ \vdots & \ddots & \vdots \\ k_{4,1} & \cdots & k_{4,6} \end{bmatrix} * \begin{bmatrix} x_1 \\ \vdots \\ x_6 \end{bmatrix}. \tag{3.20}$$

Given the fact that both angles and angular rates for pitch, roll and yaw were measured by sensors, it was assumed that all elements of the state vector were available. Substituting (3.21) into a general state-space equation yielded an equation for the closed-loop system:

$$\dot{x} = A * x - B * K * x \tag{3.21}$$

Evaluating the characteristic equation

$$det[s * I - (A - B * K)] = 0 \tag{3.22}$$

resulted in an 6th-order polynomial containing all the gains of K. The gains k_i were chosen in a way that the roots of the closed loop system (3.21) were located in the left half plane (stable system). The required elements of K were then obtained by matching desired poles to the coefficients of equation (3.22).

3.3 Optimal control or LQR design

Instead of picking root values for the closed loop system on a trial and error basis, designing a controller K and evaluating the results, it would be preferable to find the best possible solution for the gain values that produce a desired control effort and system response. A very effective and widely used technique in modern linear control systems to do this is the method of an optimal linear quadratic regulator or LQR. The basic idea is to minimize a performance index,

$$J = ½ * \int_{t_0}^{\infty} [x^T * Q * x + u^T * R * u] dt, \tag{3.23}$$

where matrix Q penalizes the state and matrix R penalizes the control effort. To get the optimal values for K which places the closed-loop poles at the stable roots of the symmetric root-locus equation the algebraic Riccati equation,

$$A^{T*}P + P * A + Q - P * B * R^{-1} * B^T * P = 0 \tag{3.24}$$

was solved to obtain matrix P. Knowing P, the gain matrix K could be calculated with

$$u = -K * x = R^{-1} * B^T * P * x. \tag{3.25}$$

Optimal control design has the advantage of speeding up and simplifying the tuning process to find the desired gain values. It limits the parameters one can influence to the two matrices R and Q, and in theory and simulation results in the best solution for a control problem.

4. Testing and Results

This section will summarize the testing and results of the two designed controllers by using MATLAB/Simulink simulation, hardware implementation via MATLAB/Simulink and via embedded C code.

4.1 MATLAB/Simulink Simulation

The dynamical quadcopter model of the prior section and the derived control algorithms were implemented in MATLAB/Simulink to illustrate that it was possible to control the attitude of the quadcopter. The model was created with the identified aircraft parameters. The testing environment contained both the linearized model for validation purposes as well as the nonlinear dynamical model of the quadcopter to simulate real world conditions and behavior.

In a first simulation the linearized model was used for a simulation time of 5sec and an initial offset of pi/4 rad/s on pitch, roll and yaw angle. The initial velocities were all assumed to be 0 m/sec. The desired state for each angle as well as the angular rates was the quadcopter's hovering condition where all angles and velocities are zero. The simulation results are shown in figure 4.

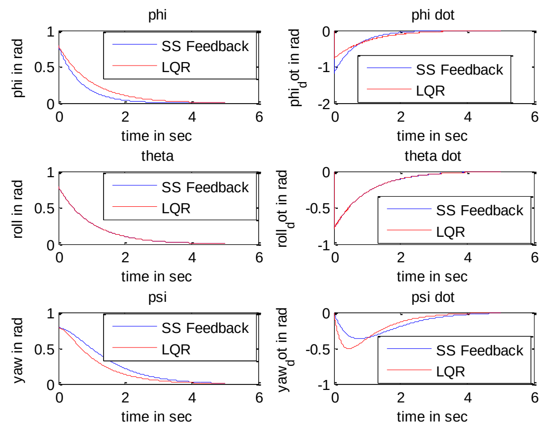

Figure 3. Nonlinear model simulation results

Both controllers performed as expected on the linearized model and were able to compensate for the initial disturbances without any problems. When used with the nonlinear dynamical model the LQR control architecture clearly had a theoretical advantage over any state-feedback controller. It reduced the overshoot for the angular velocities and provided a very good settling time of less than 2s for minor disturbances.

4.2 MATLAB/Simulink hardware implementation

The first approach to implement the control architectures on the quadcopter hardware used various blocks from Mathworks MATLAB / Simulink Embedded Targets library which required an Embedded Coder™ license. It supported Texas Instruments "TMS320F28335" as well as many other microcontroller models. Embedded Coder™ provides MATLAB / Simulink with the ability to generate readable, compact, and fast C and C++ code for use on embedded processors, on-target rapid prototyping boards, and microprocessors used in mass production. Embedded Coder™ can establish a direct connection between the MATLAB environment and the hardware for design and testing purposes. The entire quadcopter required at least four basic MATLAB / Simulink modules to be implemented, a wireless module to establish a communication between the aircraft and a ground control station, the ground control itself which in this case consisted of a joystick module, a ADC input module to read the required sensor signals for the controller and a PWM output module to connect the control board to the motors of the aircraft.

Unfortunately, Embedded Coder™ only worked with major restrictions and did not support Automation Interface and Processor-in-the-loop communications with CCSv4, which proved to cause many problems for a fully functional implementation. It is important to note that at the time of this work, Mathworks was planning to fully support the next generation of TI's Code Composer Studio 5, including a hardware-in-the-loop design and testing environment.

4.3 Embedded C code hardware implementation

Texas Instruments Code Composer Studio 4 software was used to develop code, which was then downloaded to the board via a USB/JTAG XDS100v1 module. As of the time of writing, the CCSv4 license for the XDS100-series emulators was free, so it provided a cost effective alternative to the MATLAB / Simulink Embedded Coder toolbox. TI provides an extensive library for the TMS320F28335 processor including many important code snippets as well as some useful demo programs to experiment with.

The code contained two major components that were necessary to implement the afore mentioned control architecture: a signal processing component which calculated estimates for pitch, roll and yaw and their derivatives, and a controller that adjusted the actual motor output signal based on sensor values and calculated gains.

The sensor information from the accelerometer and gyroscopes was first cleared of any bias to introduce positive and negative values before it was multiplied by a scaling gain. Then the accelerometer values were low-pass filtered while the gyroscopic values went through a high-pass filter. Finally, both were fused together to get an estimate for pitch, roll and yaw as well as their respective time derivatives.

The control part was a rather straight forward implementation of the gain matrices derived in the prior chapter. At first the error variables for pitch, roll and yaw as well as their discrete time derivatives were calculated. These, were then multiplied by the respective P and D gains for each motor and added to the throttle command signal. It was taken into account that a transformation of these gains from ω to PWM was required.

The control architecture was implemented and running on the quadcopter but didn't perform as well as in the simulation. The lack of any hardware-in-the-loop functionality to get an idea of the actual sensor signals made it impossible to narrow down the actual problem. Data acquisition for sensor signals and motor output signals of the quadcopter will be an issue for future work. It was predicted that there were several issues that caused the lack of acceptable performance when the control architectures were tested on the quadcopter, including: sensor noise, wrong gain values from RPM to PWM output signals, inaccurate parameters, lack of accounting for output signal delays (motor dynamics were not modeled) or uncertainties in the kinematic model.

5. Conclusions and Future Work

In conclusion, this work has presented the methodology to implement several control architectures on a quadcopter when there is limited information about the dynamic kinematic model. It has summarized a method to identify the parameters, derive the kinematic model, simulate the controllers and implement them on the hardware. The results indicate that although the simulations performed well, it is necessary to have a more accurate model, with better sensor data and a more robust controller in order to obtain stable flight on the actual hardware. Some of this could be achieved by analyzing sensor data, repeatable controller tuning and hardware-in-the-loop methods.

References

[1] S. Bouabdallah, P. Murrieri, and R. Siegwart, Design and Control of an Indoor Micro Quadrotor, *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*, *5*(26), New Orleans, LA, April 26 - May 1, 2004, 4393-4398.

[2] B. Erginer and E. Altug, Modeling and PD Control of a Quadrotor VTOL Vehicle", *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium (IV'07)*, Istanbul, Turkey, June 13-15, 2007, 894-899.

[3] S. Bouabdallah and R. Siegwart, *Towards Intelligent Miniature Flying Robots*, (Berlin, GE: Springer, 2006).

[4] P. Pounds, R. Mahony, P. Hynes, and J. Roberts, Design of a Four-Rotor Aerial Robot, *Proceedings of the 2002 Australasian Conference on Robotics and Automation*, Auckland, Australia, November 2 - 29, 2002.

[5] P. Pounds, R. Mahony, Corke, P. Corke, and J. Roberts, Towards Dynamically-Favourable Quad-Rotor Aerial Robots, *Proceedings of the 2004 Australasian Conference on Robotics & Automation (ARAA)*, Canberra Australia, December 6 - 8, 2004.

[6] A. Tayebi and S. McGilvray, Attitude Stabilization of a VTOL Quadrotor Aircraft, *IEEE Transactions on Control Systems Technology, 14*, May 2006, 562-571.

[7] T. B. Burchett, "Alternative Derivation of the Euler Axis Kinematic Differential Equation Using Eigensystem Derivatives," *Proceedings of the 2006 AIAA Atmospheric Flight Mechanics Conference*, 2006.

[8] H. Voos, "Nonlinear and Neural Network-based Control of a Small Four-Rotor Aerial Robot," *IEEE/ASME international conference on advanced intelligent mechatronics*, September 4 - 7, 2007, 1-6.