12-2008

# Development Of A Quadrotor Testbed For Control And Sensor Development

Abhishek Bhargava
*Clemson University*, mailbox.abhi@gmail.com

# DEVELOPMENT OF A QUADROTOR TESTBED FOR CONTROL AND SENSOR DEVELOPMENT

---

A Thesis
Presented to
the Graduate School of
Clemson University

---

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

---

by
Abhishek Bhargava
December 2008

---

Accepted by:
Dr. Timothy Burg, Committee Chair
Dr. Darren Dawson
Dr. Stan Birchfield

## ABSTRACT

A quadrotor is an under actuated unmanned aerial vehicle (UAV) which uses thrust from four rotors to provide six degrees of freedom. This thesis outlines the development of a general purpose test bed that can be used for sensor and control algorithm development. The system includes the means to simulate a proposed controller and then a hardware in the loop implementation using the same software. The test bed was assembled and verified with a linear controller for both attitude and position control using feedback from an IMU (Inertial measurement Unit) and a Global Position System (GPS) sensor.

The linear controller was first implemented as a PID controller which attempts to control the attitude of the quadrotor. The controller was simulated successfully and then experiments were conducted on a DraganFlyer X-Pro quadrotor to verify the closed loop control. The experiments conducted checked the response of the quadrotor angles to the commanded angles. The controller gains were tuned to provide stable hover in all three angles.

The Videre stereo vision system was investigated as a sensor to estimate height of the UAV above the ground. Experiments were performed that show that show static (no motion of the camera) estimates over the range 0.5 - 4 meters. The accuracy of these measurements suggest that the system may provide improved height estimation, over WAAS corrected GPS. A means to add this sensor into the UAV test bed is discussed.

## DEDICATION

I dedicate this to my family for their support and encouragement throughout.

## ACKNOWLEDGMENTS

I would like to acknowledge my committee and my professors for all their help and advice throughout my master's degree. I would like to acknowledge the help and support of Dr. Gunther Dorn for his extensive advice on the controller and the quadrotor system; Peng Xu for his help with tuning the PID gains and performing various tests on the quadrotor; Dongbin Lee for all the time he spent reading my thesis and making corrections along with all his advice on control systems; Dulu Shu for the time he spent helping me with circuits and diagrams. Their help and support is invaluable to this thesis being completed.

TABLE OF CONTENTS

Table of Contents (Continued)

LIST OF TABLES

LIST OF FIGURES

List of Figures (Continued)

List of Figures (Continued)

# CHAPTER ONE

# INTRODUCTION

## Background

A quadrotor is a four rotor helicopter in which relative speed of the four fixed-pitch blades is varied to create motion. The first full-scale four rotor helicopter was built by De Bothezat in 1921 [1]. In this work, the quadrotor will be developed as an UAV (Unmanned Aerial Vehicle) which does not require an onboard pilot. UAVs are typically airplanes and helicopters which have been equipped with computer control in order to perform autonomous or semi-autonomous tasks. Recently, there has been considerable interest in research related to UAVs due to increased affordability, increased payloads and advances in technology such as higher energy density Lithium Polymer batteries, more accurate sensors, and more efficient motors. Applications of UAVs include surveillance, target acquisition, search and rescue, meteorology among many others where a smaller vehicle is required, risk to a manned flight is too great, lower cost of operation is needed, and stealth is mandated [1].

Traditional helicopters and quadrotor helicopters have different flight dynamics. The main difference is the manner in which each compensates for gyroscopic torques. A traditional helicopter uses two rotors, the main rotor and one tail rotor, to control attitude and height. The tail rotor is used to compensate for the yaw torque generated by the main rotor and to yaw the aircraft directly.

Quadrotors are under-actuated six degree-of-freedom rigid-body vehicles that use the differences in speed of the four rotors to achieve a desired orientation and/or position. The blades have constant pitch and rotate in only one direction. The spinning directions of the blades are set in pairs to balance the torques, eliminating the need of a tail rotor for compensation. The differences in speed of the four rotors create rotational motion about the roll, pitch and yaw axes while thrust in the upward direction is a sum of the forces produced by the four rotors. The quadrotor is able to translate up and down while rotating about the three axes. The other two horizontal translations are effected by coupling the orientation of the UAV with the thrust force.

<u>DraganFlyer X-Pro</u>

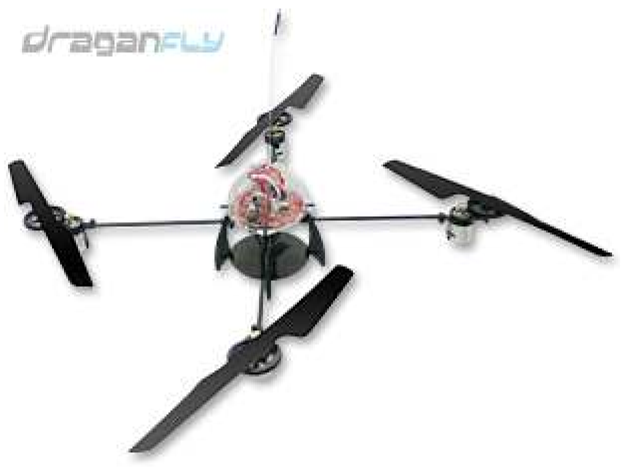


Figure 1.1 DraganFlyer X-Pro from RC Toys

The quadrotor used in this project is a DraganFlyer X-Pro [9] which is R/C quadrotor acquired in 2005 by the Clemson UAV Laboratory. It is commanded using four inputs, for thrust, roll, pitch and yaw, by the pilot using a standard hobby radio control unit. This quadrotor has four brushed DC motors (RS-545SH-5018) which provide torque for the four blades. The motors have an operating voltage between 4.5-12 volts [14]. The quadrotor has three gyros onboard which are used by the internal control loop to stabilize the quadrotor in flight. There is no information on how the internal control loop works other than the qualitative description that it acts to dampen angular motion. The signals to the quadrotor are sent using a JR PROPO PCM9XII R/C controller working at 72 MHz. The throttle, roll, pitch and yaw signals are received on the quadrotor using an R/C receiver. The internal control loop uses these signals to send PWM signals to the four motors using IRL1404 power FETs [15]. These FETs did not originally come with the quadrotor but due to overheating issues, the old FETs were removed and the new ones put in. The X-Pro quadrotor will be modified to serve as the airframe of UAV testbed.

## Sensing

For position control of any UAV, it is necessary to measure the actual position and orientation. Global positioning systems (GPS) are widely used for finding the translational position and velocities with respect to earth using satellites. However, due to interference caused by atmospheric disturbances, electromagnetic interferences, multi path errors and depending on the quality of the receiver, the GPS can have an undesirable error in the signal. These errors can range from anywhere between 5-15 meters on average GPS receivers. Such errors can be reduced by using better receivers, using DGPS (Differential Global Positioning System) as a reference and/or combining the GPS sensor data information with data acquired through other sensors. Other position sensors include ultrasonic range finders, IR (Infrareds) range finders, electromagnetic sensors and cameras for feature tracking and distance estimation. The obvious disadvantage of these sensors versus GPS is that they only provide a relative position of the UAV.

Differential Global Positioning System (DGPS) enhances the accuracy of the actual GPS signal by using a network of ground based reference stations that broadcast the difference between the positions indicted by the satellites and the known fixed positions of the stations. The broadcast of the difference in position is used at the receiving station to correct the position indicated by the sensor. A widely used variation of this approach is called Wide Area Augmentation System (WAAS) which is a Space Based Augmentation System (SBAS) in which the correction is sent via a regular satellite channel. Ultrasonic range sensors have a drawback of medium to large errors due to reflections of surfaces and thus are generally used in conjunction with other sensors including GPS for position estimations. Similarly with IR sensors, reflections can cause unwanted errors in the estimate and are generally used in combination with other sensors. Vision has been used by a number of researchers for their experiments on flight stabilization [5] [13] [16] [18] [23]. Feature tracking, optical flow and visual servoing using onboard cameras has shown that vision results can be used independent of other sensor data for position estimation.

Orientation of an UAV can be determined using micro electromechanical systems (MEMS). Such inertial navigation units normally use Kalman filtered data from rate gyros, accelerometers, and magnetometers. Integrating the rate gyros provides good attitude estimation over short period of times and accuracy is dependent on the accuracy of gyros used for estimation. Regardless of the quality of gyros used, integrating even a small error can cause unbounded error over a short period of time. For this error to be contained, secondary measurements are used in the Kalman filtering to compensate, namely accelerometers and magnetometer information to reduce the error. Commercial measurement units such as Microbotics MIDG II use both the GPS and the MEMS sensors to best estimate the six degree of freedom position and orientation of the quadrotor.

## Previous Work

Much of the ongoing research in UAVs is directed towards new methods for control, trajectory generation, and sensing. The increased applicability of UAVs in various scenarios has also made it a topic of current interest with researchers. Hamel et al [6] define the quadrotor dynamics for an X-4 flyer which is similar to the DraganFlyer X-Pro used here. This proposed dynamic model treats the quadrotor as a rigid body which can thrust and torque by itself in mid-air.

Hoffman et al describe STARMAC (Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control) [2] which uses GPS and IMU sensing to implement a control system following prescribed waypoint trajectories and to create a test bed platform for experimentation and validation of multi-agent control algorithms. Further, Hoffman et al [3] describe the use of GPS, an inertial measurement unit (IMU). and sonic ranging sensors for altitude, attitude and position estimation. The emphasis of this paper was to understand the conditions that arise when the quadrotor deviates from a hover flight for a fuller understanding of the forces that act on the quadrotor, eventually for an improved controller performance.

Researchers at Aalborg University [11] redesigned and replaced the hardware on a DraganFlyer X-Pro as access to change factory programmed inner loop controller was not available, similar to the X-Pro used herein. The inner loop controller provides the necessary torques to the motors for the orientation of the quadrotor and control of the thrust force. They simulated two controllers, a Linear Quadratic Controller (LQR) and a Piecewise Affine Hybrid System (PAHS), with the aim of autonomous flight. Another group in Aalborg University [12] did a platform development and estimation of the DraganFlyer X-Pro with the aim of autonomous flight. Both groups did extensive modeling of the motors, rotors, the body and the sensors. The estimators [12] designed by the group utilized steady state Kalman filters and an unscented Kalman filter for estimation due to differences in the sampling frequencies of the sensors. However, the estimator was not tested in actual flight.

Researchers at Brigham Young University [13] use a GPS denied indoor settings as motivation to build a vision assisted velocity and position estimator to autonomously control a quadrotor. Using a test stand, they checked the validity of the vision system to estimate heading. A linear controller was implemented with the vision system. This controller is similar to the linear controller utilized in this work. The controller allows for position, velocity and orientation to be used as feedback to autonomously control the quadrotor.

Jong and Tamlin [4] use a single GPS as the only sensor of vehicle position and velocity for closed loop autonomous control of a UAV. MITs RAVEN (Real-Time Indoor Autonomous Vehicle Test Environment) paper [5] uses a global metrology motion capture system for indoor analysis and experimentation with multiple autonomous UAVs. The systems uses a number of autonomous micro UAVs to autonomously track a vehicle on the ground, again in a GPS denied environment.

Park et al [16] use a an embedded controller with feedback provided by an INS (Inertial Navigation System) using three rate gyros and three accelerometers, a CCD (Charge Coupled Device) camera with wireless communication transmitter for observation and an ultrasonic range sensor for height control. They used a RIC (Robust Internal Loop Compensator) based disturbance compensation and a vision based localization method to get the quadrotor to perform stable flight. Other than the ultrasonic range sensors, IRs (Infrareds) were also used to avoid obstacles.

P.Castillo et al [17] performed autonomous take off, hovering and landing control on a QRT (Quadrotor type) UAV by using a Lagrangian model and Lyapunov based control approach with orientation and translational being measured by a Polhemus 3-D tracker system. The Polhemus uses electromagnetic measurements from sensors attached to the quadrotor and read them via sensors around the room. Real time experiments were performed on a DraganFlyer micro quadrotor similar to the X-Pro used here.

Researchers at Autonomous Systems Lab at the Swiss Federal Institute of Technology in Lausanne, Switzerland developed the OS4 quadrotor. The quadrotor based the research on attempting autonomous flight using vision [18]. The OS4 used IMUs

(Inertial Measurement Units) and a PID control structure for stabilization. Vision was used for controlling drift and ultrasonic sensors were used for height control. An integral back stepping approach was added for better altitude control and cascaded in to the PID control structure.

## Outline of the UAV Testbed Development

This thesis is divided into three main parts to follow the stages and milestones in the project. The first step in designing and building the UAV testbed was to create the software environment that can be used for both simulation and hardware in the loop experiments. An overview of the system is shown in Figure 1.2. The first part describes the use of QMotor [27] software to incorporate a mathematical system model of a quadrotor, hooks for control algorithms, hardware interface to the X-Pro helicopter and sensor inputs. A linear controller is used to demonstrate the simulation capabilities of the system. The reason for using a linear controller was to easily implement the vision system with the position controller, with the vision system estimating heights close to the ground. The combination of the vision system and GPS for height estimations would have been done easier where small angle approximations are done, as in the position controller proposed in this thesis. The controller takes as input a desired position, specified by the user, and uses it to generate desired velocities and desired angles. The controller uses feedback of position, velocity and orientation of the quadrotor.

Using the quadrotor system model, simulations are shown for both angular and position control. The height and the attitude of the quadrotor can be directly actuated using the inputs. However to achieve the remaining translation motions, thrust and orientation of the quadrotor are coupled. It is useful to think of the linear controller as having an inner angle controller, wrapped by an outer loop position controller. Thus for any control of the 3D position of the quadrotor, it is necessary to first attitude control. First simulations were carried out for the angular controller to check the behavior of the model with the proposed angular controller. The position controller was added once the attitude controller gains had been properly tuned. The position controller is an extension of the attitude controller, as translational forces are dependent on the orientation of the quadrotor for x and y axis direction. Results and conclusions are given at the end of chapter 2 relative to this work.

The second phase of the project was to perform experiments using the proposed linear attitude and position controllers and the QMotor software. The setup and the equipment used for the testing of the quadrotor are described first in Chapter 3. The X-Pro is modified to carry an IMU (Inertial Measurement Unit) and GPS to support autonomous stabilization and control of the orientation and position of the quadrotor using the position, velocity and orientation information provided by the sensors.

Attitude control is achieved by getting the quadrotor to autonomously hover, with thrust given manually using a joystick slider as input. Various other tests were performed to check the response of the system to desired orientation requirements. Once the attitude controller is implemented, the next step was to test the position controller using the values attained from the attitude controller. However, position control was not achieved at the writing of this thesis due to issues with sending correct throttle commands to the

UAV, GPS sensor error reading (Appendix) and varying motors responses with change in applied voltage. Experiments are still ongoing in an effort to completely test the position controller.

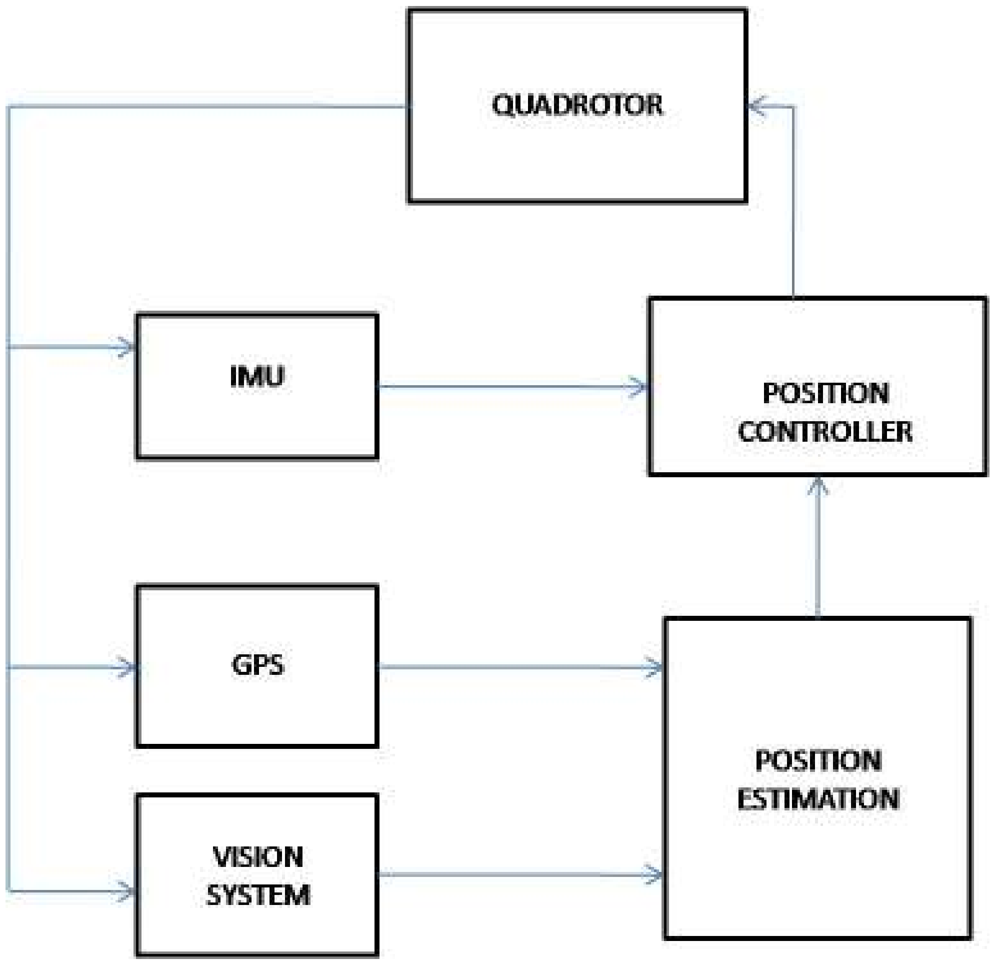


Figure 1.2 System Overview

One of the reasons for building a new open UAV platform was to test new sensors. The first new sensor to be considered was a stereo vision system to measure the height of the quadrotor under low altitude hover conditions. Due to inaccuracies in the

GPS data (Appendix), the vertical error can be anywhere between 5-15 meters. Hence under conditions, where the quadrotor is close to the ground, a new sensor is needed. For this, an STOC-DCSG (Stereo on Chip) camera manufactured by Videre Systems [19] was to be used separately to check the distance of the quadrotor from the ground. Several tests were performed to check the accuracy of the distance estimated by the camera for several surfaces. The distance estimation was to be done using SVS (Small Vision System) [20] software provided along with the camera. The software allows real time performance and thus information of actual height up to a certain distance can be gauged using the camera. However due to hardware problems with the camera system itself, the camera was not used as an additional sensor on the quadrotor. Results of the tests are still included, as a basis for future work that can be performed using the vision system.

The final chapter compares the results achieved using the simulations with the results achieved with the experiments. It also contains the recommendations for future work, along with the changes that are required for better overall control of the system.

# CHAPTER TWO

# PROPOSED CONTROLLER AND SIMULATIONS

## Introduction

A quadrotor is an UAV (Unmanned Aerial Vehicle) in which considerable research interest has been shown due to advancement in technology, affordability and increased applicability [7]. It is an under actuated vehicle that has four inputs and six degrees of freedom. Due to four inputs, two degrees of freedoms are achieved by combining thrust and orientation of the quadrotor. Due to instability of the quadrotor, we have to use a controller to achieve any desired position or orientation.

As said earlier, the quadrotor is an under actuated system with six degrees of freedom and four inputs. The quadrotor is typically modeled as a rigid body that thrust and torque freely in mid air. Any position or orientation of the quadrotor is achieved by changing the torque generated by the four rotors, which is further achieved by changing the relative speed of the four rotors. The changes in relative speed can create roll, pitch and yaw in quadrotor body axes while thrust is an addition of the torques produced by all the four blades. The thrust only allows for quadrotor motion in up or down direction, while to achieve the other two translational motions, thrust coupled with pitch and/or roll of the UAV are used.

This chapter is divided into six sections, with the first section being the introduction. The second section contains basic information about the DraganFlyer X-Pro and gives an overview of the forces acting when inputs are given. The third section illustrates the co-ordinate frames used. The fourth sections contain the quadrotor dynamics and kinematics. The fifth section covers the controller proposed for the attitude and position control of the quadrotor.

The next section covers simulations for the attitude and position controller. The attitude controller is simulated with desired angles given using a joystick input and the response of the system is seen. The position controller is simulated using a desired trajectory generated using a simple ramp function and actual position as input for commanded position. The start and end points of the quadrotor are defined for the position controller. Here the quadrotor dynamics are used instead of the sensors to provide feedback to the controllers. The results from the simulations of the attitude and position controllers with conclusions of the results are shown.

## Overview of Quadrotor Motion

The DraganFlyer X-Pro [9] shown in Figure 1.1 has four rotors which can independently spin in one direction at varying speeds to orient and position the aircraft. Figure 2.1 shows that the rotor of the front and rear of the aircraft turn in the counter clockwise direction using a left-hand pitched blade to create vertical lift while the rotors at the side of the aircraft turn in the clockwise direction and use a right-hand pitched blade to produce lift. Note that none of the rotor can reverse direction and produce negative thrust. The rotors spinning together at a constant speed allow the quadrotor to maintain a stable hover; that is, each of the rotors produce $\frac{1}{4}$ the force needed to counteract the gravitational pull as shown in Figure 2.2. If the speeds of the rotors are simultaneously increased, providing more thrust, the quadrotor will rise up. Similarly, if speeds of the rotors are simultaneously reduced, the overall thrust is reduced and the quadrotor will settle.

The rotors are controlled in pairs to produce speeds in roll, pitch and yaw. The rotors are thus grouped into two sets, I and II, which spin the in the same direction as shown in Figure 2.1. Set I contains the rotors 1 and 3 which spin in the counter-clockwise direction. Set II contains rotors 2 and 4 which spin in the clockwise direction.

The quadrotor orientation is defined by the roll, pitch and yaw angle as shown in Figure 2.1. Pitch is defined as rotation about the y axis, roll is defined as rotation about the x axis, and yaw is defined as rotation about the z axis with positive directions as shown in figure 2.2. In the diagrams, black indicates normal speed, dark blue indicates increased speed and white indicates reduced speed of that rotor.

Pitch is adhered by changing the relative speed of the rotors within set I while maintaining the speed of set II. For positive pitch, as shown in Figure 2.3, rotor 1 must speed up and rotor 3 slows down. This difference in rotor speeds in set I means that rotor 1 is now producing more thrust than rotor 3; these unbalanced thrusts create a torque about the x axis that acts to pitch the aircraft.

In order to create a negative pitch, the opposite happens to rotor set I. Rotor 1 slows down and rotor 3 speeds up, with speed of rotors of set II remaining constant. To maintain the same total level of thrust, the increase in speed of the rotor of one motor is equivalent to the decrease in speed of the rotor of the other motor of the set I, maintaining the same thrust level overall. Note that when the quadrotor pitches, the net direction of

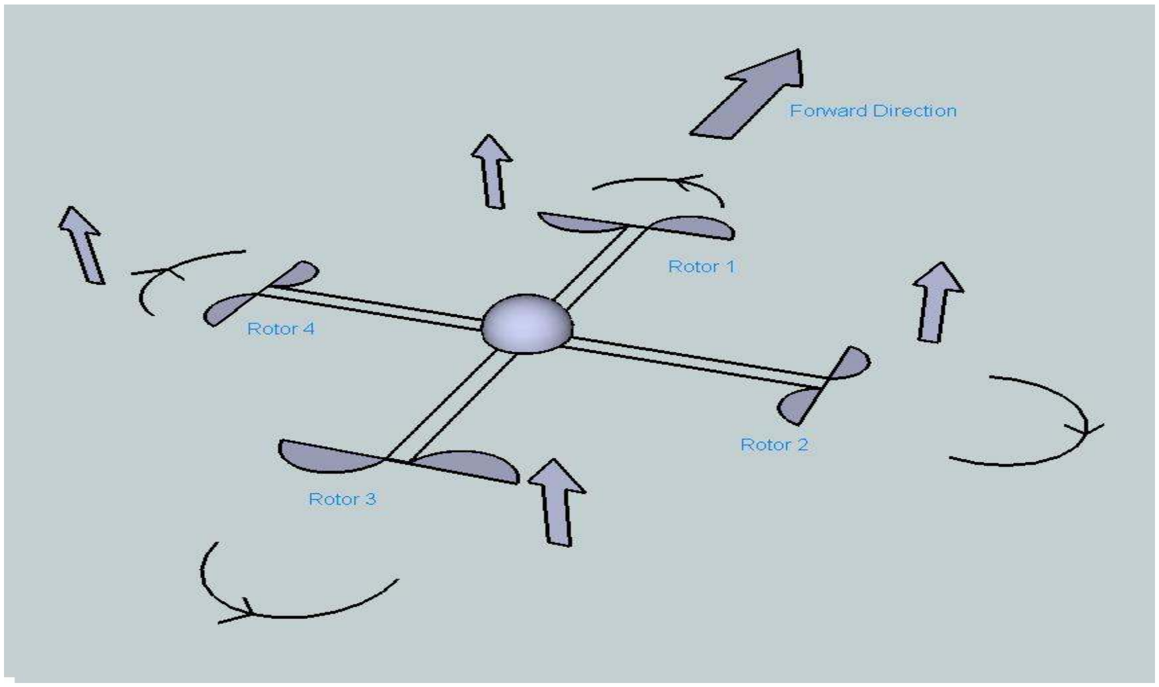


Figure 2.1 All four rotors spinning at same speed to provide upward thrust

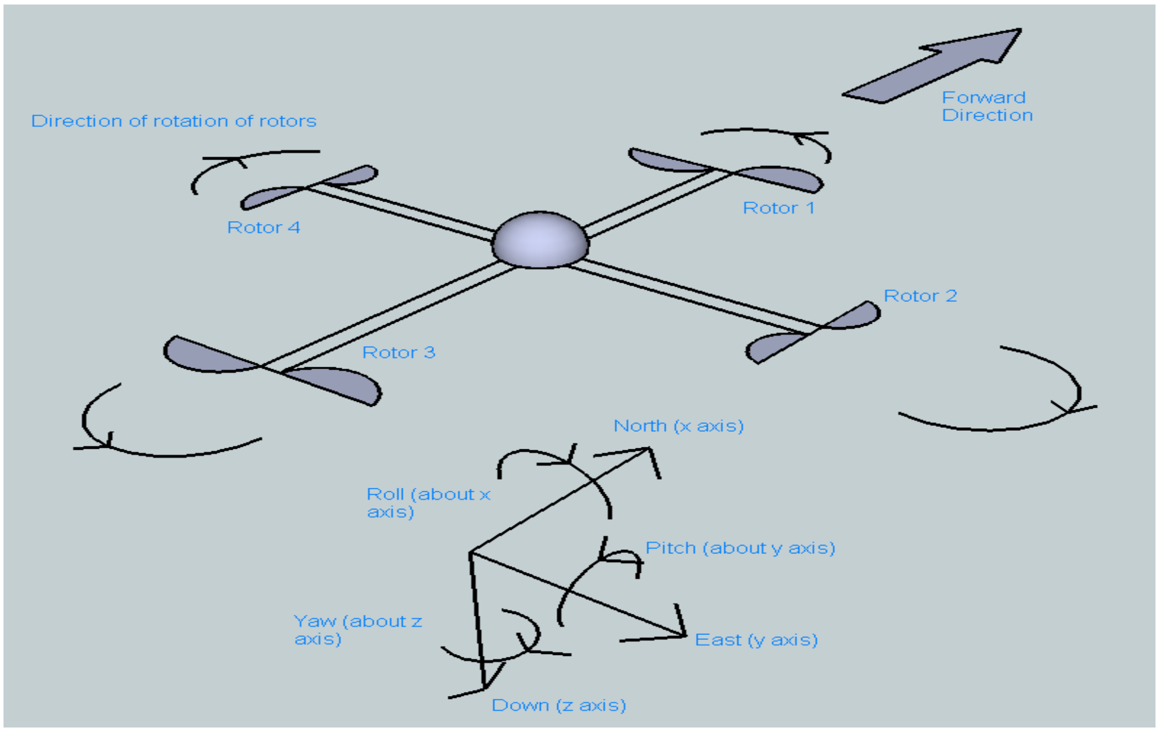


Figure 2.2 Yaw, pitch and roll definitions along with direction of rotor motion

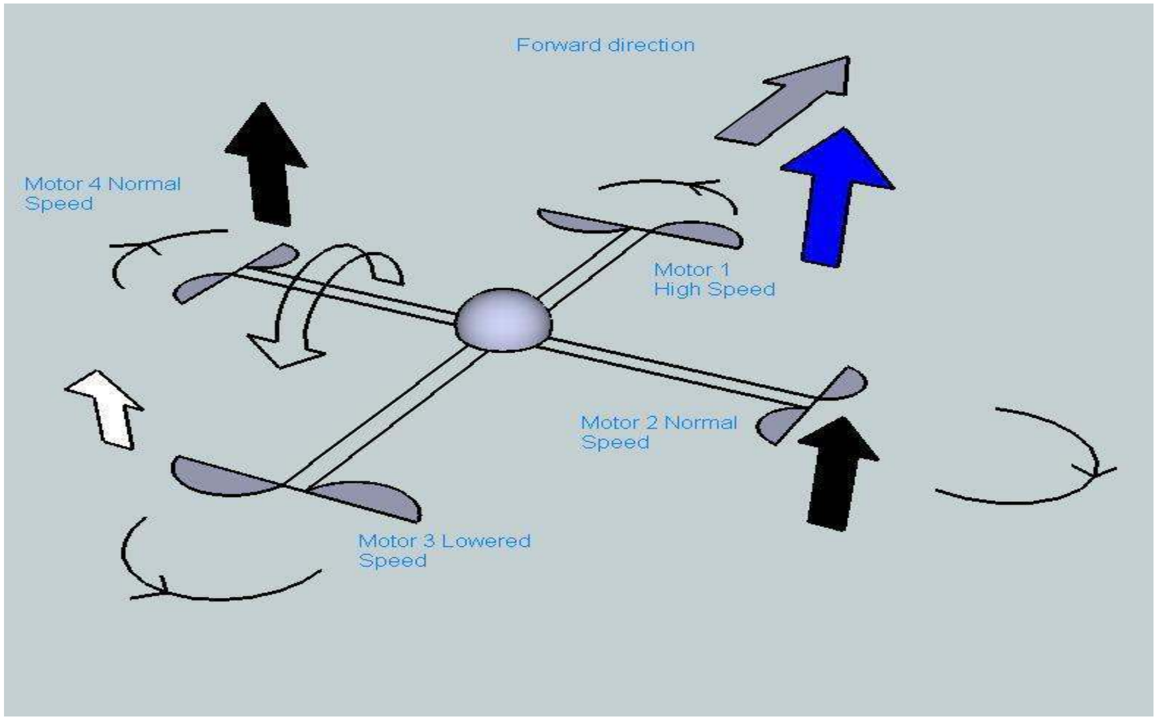


Figure 2.3 Positive Pitch

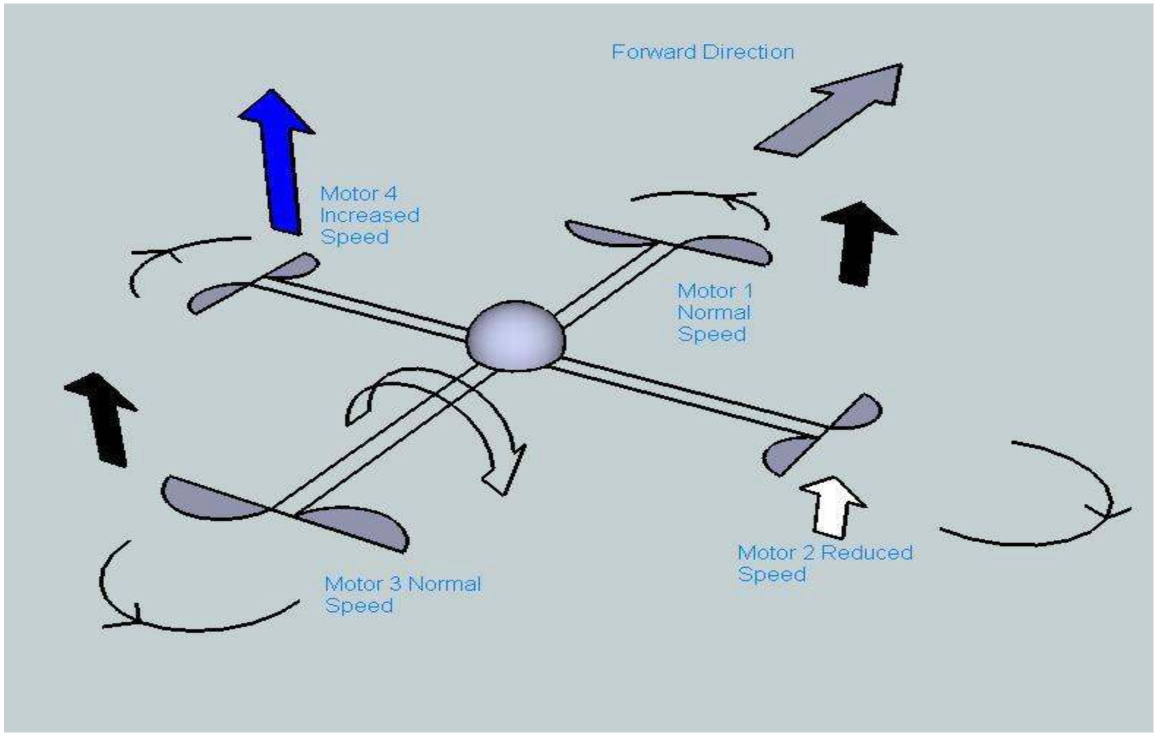


Figure 2.4 Positive Roll

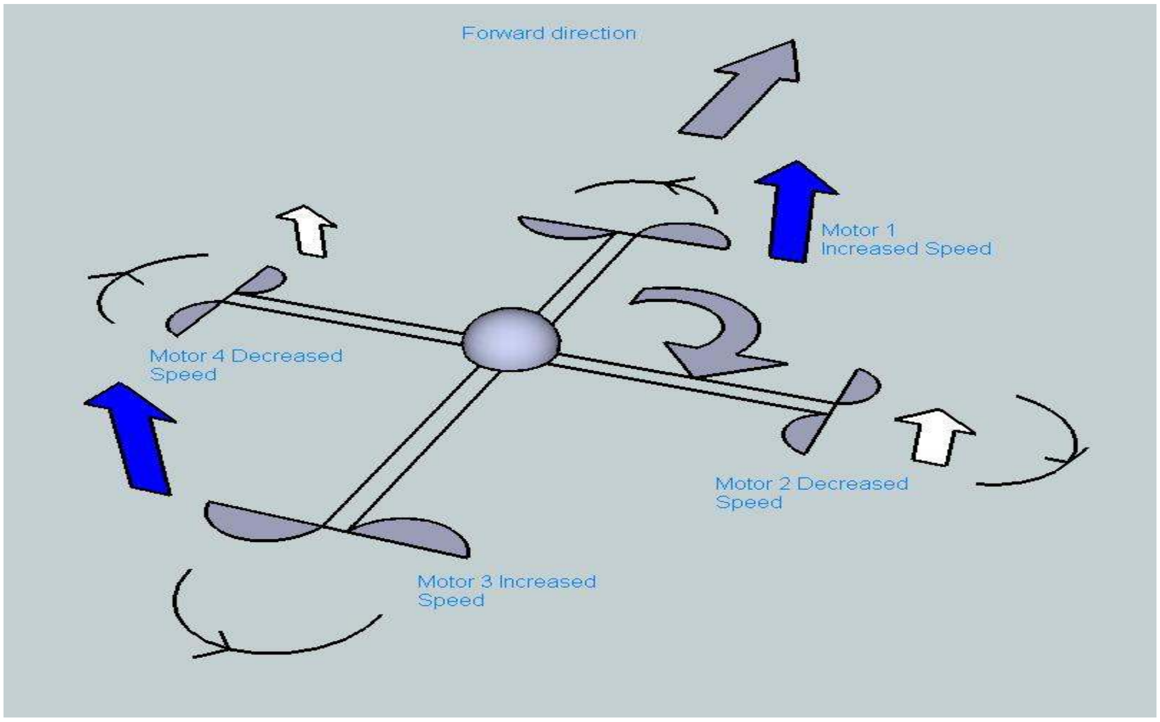


Figure 2.5 Positive Yaw

thrust force is no longer pointing in the z direction, reducing the overall thrust provided in z direction. Therefore, the quadrotor settles unless more thrust is provided to compensate.

Roll is rotation about the y axis and is initiated by changing the speed of rotor set II. For a positive roll, shown in Figure 2.4, the rotor 2 must decrease in speed and rotor 4 increases in speed, at the same time maintaining the speed of the rotor set II. Similarly for a negative roll, rotor of motor 2 speeds up and rotor of motor 4 slows down. Again for maintaining the same net level of thrust, the decrease in force from one rotor is equivalent to the increase in force of the other rotor.

For yaw, all the rotors of sets I and II are used simultaneously. Each motor rotor set creates a reaction torque as the motor turns the rotor. Each reaction force can be considered at the quadrotor center of mass. Rotors of set 1 motors spin counter-clockwise , producing a body torque in the clockwise direction and rotors of set II motors spin clockwise, producing a torque in the counter-clockwise direction. Thus set I creates a clockwise motion and set II creates a counter-clockwise motion. If all the rotors of all sets spin at the same speed, creating equivalent torque, the clockwise and counter clockwise torques cancel each other out, preventing any yaw motion as is in the case of Figure 2.2. For a positive yaw as shown in Figure 2.5, rotors of set I speed up and rotors of set II slow down, creating a net clockwise motion. At same time, if net thrust is to be maintained at the current level, the decrease in force due to set II motors is compensated by equivalently increasing the force created by set I rotors.

Three orientations have been discussed above, where it was shown that the orientations of the quadrotor can be achieved by changing the relative torques generated by the four motors individually or in groups. The three translationals of the quadrotor are x, y and z and are achieved by coupling thrust and orientation. If the quadrotor needs to move in an up or down in the z-direction, the thrust of the quadrotor is increased or decreased respectively. If all the rotors generate enough thrust upward to exactly counteract the force of gravity, the quadrotor will hover at its position. To move up, the force generated by the rotors is increased. Similarly, to move down, the net force generated is reduced and the quadrotor will settle.

For motion in x and y direction, thrust and orientation must be combined. For a quadrotor to move in the y direction, the quadrotor will need a positive roll about the x axis. Due to a component of thrust force being now directed towards the y axis direction, there is a motion in the y axis. To move in the negative y-direction, a force must be generated in the opposite direction, which is done by doing a negative roll so that the component of thrust is in the opposite direction. This illustrates the coupling between the thrust and the roll angle required to achieve motion in the y direction.

Similar to the y-axis motion, for motion in the positive x axis direction, the quadrotor needs a negative pitch about the y axis. With a component of net thrust now in the x direction, there is a motion in the positive x axis direction. To move backwards, the quadrotor needs a positive pitch, so that a component of thrust now points in the opposite direction. This shows the coupling between the thrust and pitch to achieve a position in the x-direction. Again further complicating motion is the fact that due to the change in orientation of the quadrotor, the thrust component acting in the upward direction gets reduced. This causes the quadrotor to lose height and thus will settle unless the rotor speeds are increased to compensate for the loss of height. That is , height control must be included in any translational commands.

## Co-ordinate Frames

The position of the quadrotor is expressed using aeronautical standards of NED (North East Down) frame. NED frame expresses the position of the quadrotor with its x-axis pointing north, y-axis facing east and z axis facing down towards the center of the earth. The vehicle frame is thus a co-ordinate frame translating with the vehicle but remaining parallel to the world frame (inertial frame). Figure 2.6 shows the NED frame with F denoting the body frame and I denoting the inertial frame.

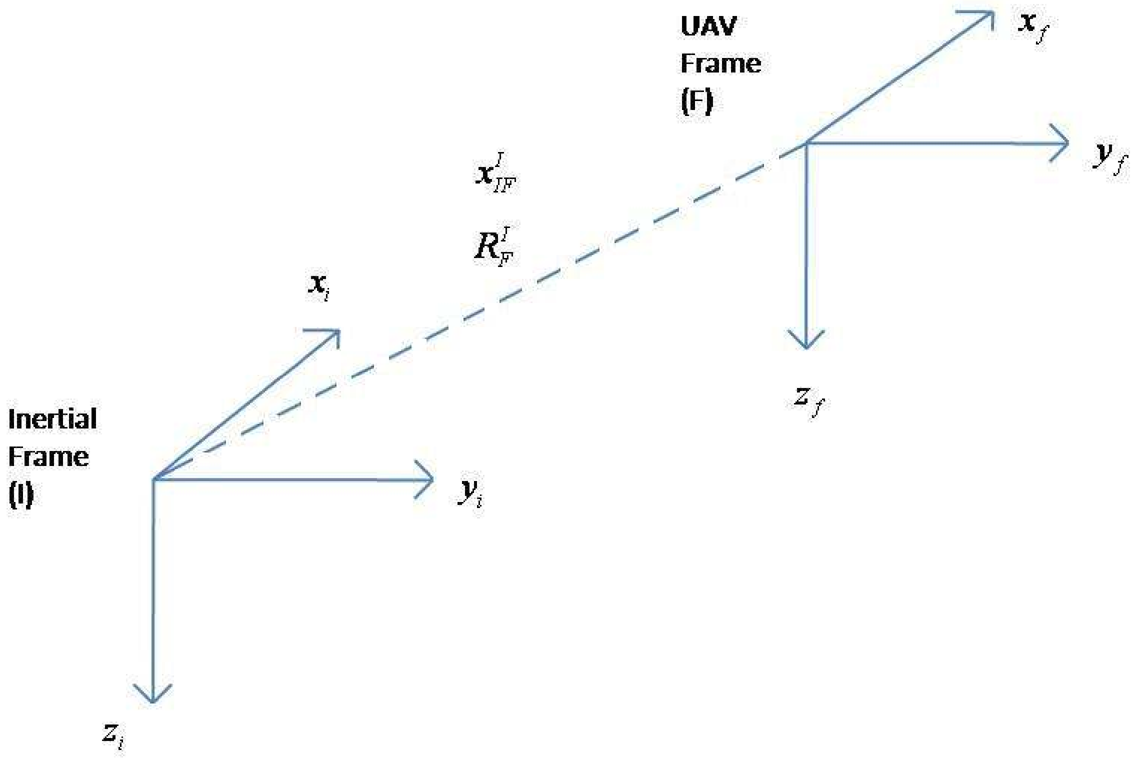


Figure 2.6 Inertial frame to Body frame using NED format

The orientation of the quadrotor is expressed using Euler angles of roll (φ), pitch (θ) and yaw (ψ) with the Euler 3-2-1 system. These three angles describe the vehicles rotation relative to three successive frames. For the vehicle frame 1, the frame is found by rotating the quadrotor about the z-axis with a positive yaw (ψ) angle. Vehicle frame 2 is found by rotating the quadrotor about the vehicle frame 1, by positive pitch (θ) angle. Vehicle frame 3 is found by rotating the quadrotor about the vehicle 2 frame, by a positive roll (φ) angle. Vehicle frame 3 is also the body frame referred to as frame F later on, with inertial frame referred to as frame I.. Quaternion's may also be used to describe the orientation of the quadrotor, but Euler angles are easier and more intuitive to deal with and the singularities affecting the Euler angles are not achieved in actual quadrotor flight.

## Quadrotor Model

### Quadrotor Dynamics

For the DraganFlyer X-Pro, the rotational torques are directly actuated and the translation torques are only directly actuated in the z direction. The forces and torques are given by

$$\begin{aligned} F_f^F &= [\ 0\ 0\ u_1\ ]^T \in \mathbb{R}^3 \\ F_t^F &= [\ u_2\ u_3\ u_4\ ]^T \in \mathbb{R}^3 \end{aligned} \tag{2.1}$$

where $F_f^F$ (t) refers to the UAV translation forces expressed in the UAV frame F and $F_t^F$ (t) refers to the UAV torques expressed in the UAV frame.

Equation 2.5 shows the rotational forces due to the torques that given to the quadrotor. Normally, a matrix [5] is used to relate the torque of each motor to the input torque as given below

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} -b & -b & -b & -b \\ 0 & db & 0 & -db \\ db & 0 & -db & 0 \\ k & -k & k & -k \end{bmatrix} \begin{bmatrix} \overline{w}_1^2 \\ \overline{w}_2^2 \\ \overline{w}_3^2 \\ \overline{w}_4^2 \end{bmatrix} \tag{2.2}$$

where $\overline{w}(t)$ are the rotor torques on the quadrotor and $d,b,k \in \mathbb{R}^1$ are constant parameters based on rotor design and placement. Equation 2.2 equates the actual rotor torques to the quadrotor force and torques from (2.1). In case of the DraganFlyer this calculation is done internally (using the inner control loop) and the joystick is mapped to $u(t)$.

Rigid body dynamics are used to describe motion of the quadrotor as it can rotate and translate freely in space as a rigid body. Hamel and Mohanys paper [6] defines the quadrotor dynamics that are given below

$$m\dot{v}_{IF}^{F} = -mS(w_{IF}^{F})v_{IF}^{F} + N_1(\bullet) + mgR_I^F e_3 + F_f^F \tag{2.3}$$

$$\dot{R}_F^I = R_F^I S(w_{IF}^F) \tag{2.4}$$

$$M\dot{w}_{IF}^{F} = -S(w_{IF}^{F})Mw_{IF}^{F} + N_2(\bullet) + F_t^F \tag{2.5}$$

Here $v_{IF}^F(t) \in \mathbb{R}^3$ is the translational velocity of the UAV with respect to the inertial frame (I) expressed in the orientation of the UAV body frame (F), $w_{IF}^F(t) \in \mathbb{R}^3$ is the angular velocity of the UAV frame, $R_F^I \in SO(3)$ is the rotational matrix that transforms a vector in UAV frame (F) to the inertial frame (I), $g$ is the gravitational constant, $m \in \mathbb{R}$ is the mass of the UAV and $M \in \mathbb{R}^{3x3}$ is the constant moment of inertia matrix for the UAV. $S() \in \mathbb{R}^{3x3}$ is a skew symmetric matrix defined using [10]

$$S(w) = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \quad \text{where } w = [w_1 \quad w_2 \quad w_3]^T \in \mathbb{R}^3. \tag{2.6}$$

$N_1 \in \mathbb{R}^3$ and $N_2 \in \mathbb{R}^3$ are the aerodynamic damping forces and moments which here are the un-modeled non-linear terms of the translational and rotational dynamics respectively. Equation 2.11 shows the time derivative of the position of the quadrotor, essentially making it the velocity of the quadrotor expressed in a different frame.

Equation 2.3 shows the acceleration that affects the body due to the thrust force applied using the rotors. The gravity term is the force that damps the motion of the quadrotor while going up and accelerates it while going down. Equation 2.4 relates the rotation matrix with its derivative using the skew symmetric matrix.

## Notation Used

The frame of reference for the position of a quadrotor will be NED (North East Down) in keeping with aeronautical standards. Angles represented herewith will be represented by their aeronautical terminology of roll (φ), pitch (θ), yaw (ψ), with roll being about the x axis, pitch being about the y axis and yaw being about the z axis.

With two or more frames of reference to be used, rotation between two frames is represented by

$$\Theta_F^I \in \mathbb{R}^3$$

where $\Theta_F^I$ are the roll, pitch and yaw angles of rotation of frame F with respect to I. Similarly, position is expressed as

$$x_{BF}^I \in \mathbb{R}^3$$

where $x_{BF}^I$ denotes the position of quadrotor in frame F relative to frame B expressed in the orientation of frame I. The position $x_{BF}^I$ can be expressed in other frames using a rotation matrix

$$R_F^I \in SO(3)$$

where $R_F^I$ is the rotation matrix used to transform co-ordinates from frame F to frame I.

## Quadrotor Kinematics

For the kinematic model, the Euler angles roll, pitch, yaw can be found using the angular velocities. A Jacobian is required to satisfy the relation between the angular velocity and angles

$$w_{IF}^{F} = J_F \dot{\Theta}_{IF}^{F} \tag{2.7}$$

which is used to solve for angles using

$$\Theta_F^I = \int_0^t J_F^{-1} w_{IF}^F dt \tag{2.8}$$

where $\Theta_F^I(t) \in \mathbb{R}^3$ represents the roll pitch and yaw angles between the UAV frame and the inertial frame. The Jacobian used in (2.8) above is defined as [24]

$$J_F^{-1} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta) \end{bmatrix}, \Theta_F^I = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{2.9}$$

To convert between the rotation matrix $R_I^F$ and $\Theta_F^I$ , the following direction cosine matrix is used

$$R_I^F = \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\theta)\sin(\psi) & -\sin(\theta) \\ \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \sin(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{2.10}$$

We also find $\dot{x}_{IF}^I(t) \in \mathbb{R}^3$, which refers to the time derivative of the position of the UAV frame (F) with respect to the inertial frame (I) expressed in the inertial frame (I)

$$\dot{x}_{IF}^{I} = R_{F}^{I} v_{IF}^{F} \tag{2.11}$$

## Proposed Linear Position Controller

While implementing a desired position controller, thrust should automatically be taken care of due to a position error created by lowered thrust. For position control of a quadrotor, z axis position is directly actuated using the thrust control, while the x and y axis positions are actuated by coupling thrust force with the orientation of the quadrotor. The proposed controller utilizes this relationship to achieve a desired position through a feedback loop which involves utilizing the sensor data given. Attitude tracking is essential to position tracking and acts as the last loop of the system [25]. Since we are working with small angles, the model can be approximately linearized.

To attain positive position in the y-axis, the quadrotor has to be given a positive roll rotation about the x axis, effectively applying a component of thrust in that direction. Due to this force, the quadrotor motion in the y axis can be continued till the orientation is maintained. To stop the quadrotor, a force in the negative y axis has to be given, applied by the quadrotor with a negative roll rotation about the x axis. This slows and eventually stops the quadrotor.

Similarly, for positive motion in the x axis, the quadrotor is given a negative pitch rotation about the y axis. Due to the component of thrust acting in the positive x direction, there is motion in that direction. The quadrotor can be stopped by giving a positive pitch rotation, which provides a thrust in the negative x direction. For positive motion in the z direction, the thrust (sum of all motor lift forces) has to be only reduced. For motion in the negative z axis direction, the thrust has to be increased.

Position and velocity control were implemented as successive loops around the attitude control as shown in Figure 2.7. Assuming that the velocity dynamics have a significantly slower response than the attitude dynamics, the attitude loops can be treated as a block with unity gain and thus the desired attitude angles are taken directly from the velocity loop outputs [13]. Similarly, the desired velocity commands can be taken directly from the position control loop outputs. Thus position error allows computation of desired velocity and velocity error allows computation of desired attitude angles. This successive loop controller is shown in Figure 2.7. Using a PID controller, the non-linearities from (2.3) and (2.5) including $N_1$ and $N_2$ will be ignored. The controller aims to achieve a desired position, by utilizing these nested control loops.

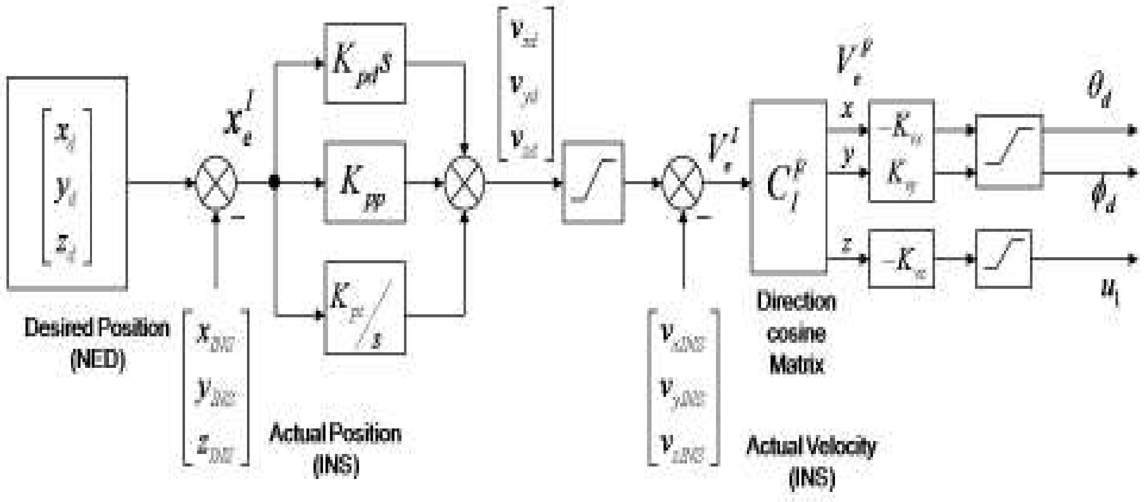
The desired velocity is attained using

$$v_d = k_{pp} p_e^I + k_{pi} \int p_e^I + k_{pd} \frac{dp_e^I}{dt} \tag{2.12}$$

where $v_d = [v_x \ v_y \ v_d]^T$ is the desired velocity, $k_{pp}$ is the position proportional gain , $k_{pd}$ is the position derivative gain and $k_{pi}$ is the position integral gain. $p_e^I$ is the error between desired and actual position given by

$$p_e^I = p_d^I - p_{INS}^I \tag{2.13}$$

where $p_d^I = [x_d \ y_d \ z_d]^T$ and $p_{INS}^I = [x_{INS} \ y_{INS} \ z_{INS}]^T$ are the desired and actual positions respectively in the inertial NED frame.



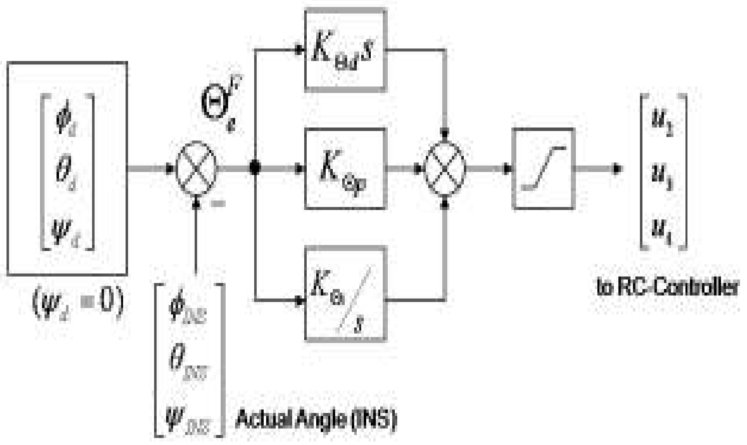


Figure 2.7 Position Controller Block Diagram

This desired velocity is saturated to make sure that it does not exceed limits and is further used to find the orientation angles and thrust using feedback for the actual velocity of the quadrotor. The velocity error is calculated initially in inertial frame using

$$v_e^I = v_d^I - v_{INS}^I \tag{2.14}$$

where $v_e^I$ is the velocity error in the inertial frame, $v_d^I = [v_{xd}\ v_{yd}\ z_{zd}]^T$ is the desired velocity and $v_{INS}^I = [v_{xINS}\ v_{yINS}\ v_{zINS}]^T$ is the actual velocity in inertial NED frame.

This velocity error needs to be converted to body frame by multiplying it with the Direction Cosine Matrix (2.10) as below

$$v_e^F = \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\theta)\sin(\psi) & -\sin(\theta) \\ \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \sin(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & \cos(\phi)\cos(\theta) \end{bmatrix} v_e^I \tag{2.15}$$

where $v_e^F$ is the velocity error calculated in the UAV frame.

The velocity in the UAV frame can be used to find the desired angles. The desired angles are then attained using

$$\theta_d = -k_{vp} v_{ex}^F - k_{vi} \int v_{ex}^F - k_{vd} \frac{dv_{ex}^F}{dt} \tag{2.16}$$

$$\phi_d = -k_{vp} v_{ey}^F - k_{vi} \int v_{ey}^F - k_{vd} \frac{dv_{ey}^F}{dt} \tag{2.17}$$

$$\psi_d = 0 \tag{2.18}$$

where $\theta_d$ is the desired pitch angle, $\phi_d$ is the desired roll angle, $\psi_d$ is the desired yaw angle, $k_{vp}$ is the velocity proportional gain, $k_{vd}$ is the velocity integral gain, $k_{vi}$ is the velocity integral gain, $v_{ex}^F$ is the error in velocity in the x direction in the UAV frame and $v_{ey}^F$ is the error in velocity in the y direction in the UAV frame.

The thrust force $F_f^F$ is calculated using the error in velocity in the z direction in UAV frame using

$$u_1 = -k_{vp} v_{ez}^F - k_{vi} \int v_{ez}^F - k_{vd} \frac{dv_{ez}^F}{dt} \tag{2.19}$$

where $u_1$ is equivalent to $F_f^F$ and $v_{ez}^F$ is the velocity error in the z direction in UAV frame.

Using $F_t^F$ (2.1) as the control signal for the angles, the PID controller uses the error between the desired and actual orientation to provide a feedback to the quadrotor based on

$$u = -k_{\Theta p} e_{\Theta} - k_{\Theta d} \frac{de_{\Theta}}{dt} - k_{\Theta i} \int e_{\Theta} \tag{2.20}$$

where $u(t) = [u_2, u_3, u_4]^T$ is equivalent to $F_t^F$, $e_{\Theta}$ is the error signal between desired and actual orientation, $k_{\Theta p}$ is the orientation proportional gain, $k_{\Theta i}$ is the integral gain and $k_{\Theta d}$ is the derivative gain. The error signal $e(t)$ is given by

$$e_{\Theta} = \Theta_d - \Theta_{INS} \tag{2.21}$$

where $\Theta_d$ are the desired roll, pitch and yaw angles and $\Theta_{INS}$ are the actual roll, pitch and yaw angles. For the simulation, we use the dynamic equations and kinematics to calculate the actual orientation while in the actual experiment the MIDG sensor was used to provide information.

## Simulations

### Quadrotor Model Parameters

For the simulation, certain parameters needed to be ascertained. The mass and inertia matrix are needed for the dynamics equations for eventual calculation of the position, velocity and angle terms. The mass of the quadrotor was checked using a digital scale, with mass of other components added on such as the RF modem and Li-Polymer batteries measured separately as shown in Table 2.1.

The torque and the forces generated by the motors were measured in a previous thesis [8]. The total force the helicopter produces was measured using a spring scale to measure the amount of force created by one rotor when spinning at maximum speed. This force multiplied by four gives the total thrust capability of the quadrotor in the z direction. For measuring the maximum yaw torque, two motors were spun at maximum speed with the other two motors were turned off, giving the maximum yaw torque. Similarly, one motor of one rotor set spinning at maximum speed with the other rotor of same set turned off gave the value for maximum roll/pitch torque. These measurements are displayed in the Table 2.1.

The inertia matrix was not measured wand was estimated from [2], where the vehicle was half the weight of the DraganFlyer, so the values were doubled to give

$$M = \begin{bmatrix} 1.3 & 0 & 0 \\ 0 & 1.3 & 0 \\ 0 & 0 & 2 \end{bmatrix} \tag{2.22}$$

Table 2.1 DraganFlyer X-Pro Parameters

| Parameter | Value | Units |
|---|---|---|
| Quadrotor Mass | 2.041 | Kg |
| Batteries + RF Modem + Sensors | 0.68 | Kg |
| Maximum Thrust | 35.586 | N |
| Maximum Roll/Pitch Torque | 4.067 | Nm |
| Maximum Yaw Torque | 2.034 | Nm |

## Simulation

To check the validity of the proposed controller, simulations were done, first checking the attitude controller and then the position controller as a whole for a generated desired trajectory. For the simulations, there are three main steps. The first step involves using the dynamics equations from (2.2) – (2.5) to calculate the actual orientation, velocity and position of the quadrotor. Secondly, those values have to be input into the controller, where values for the thrust and roll, pitch and yaw torques are calculated to achieve the desired orientation, velocity and position. These form the control input to the system. Finally, these control input values are fed back into the dynamics equations to close the loop.

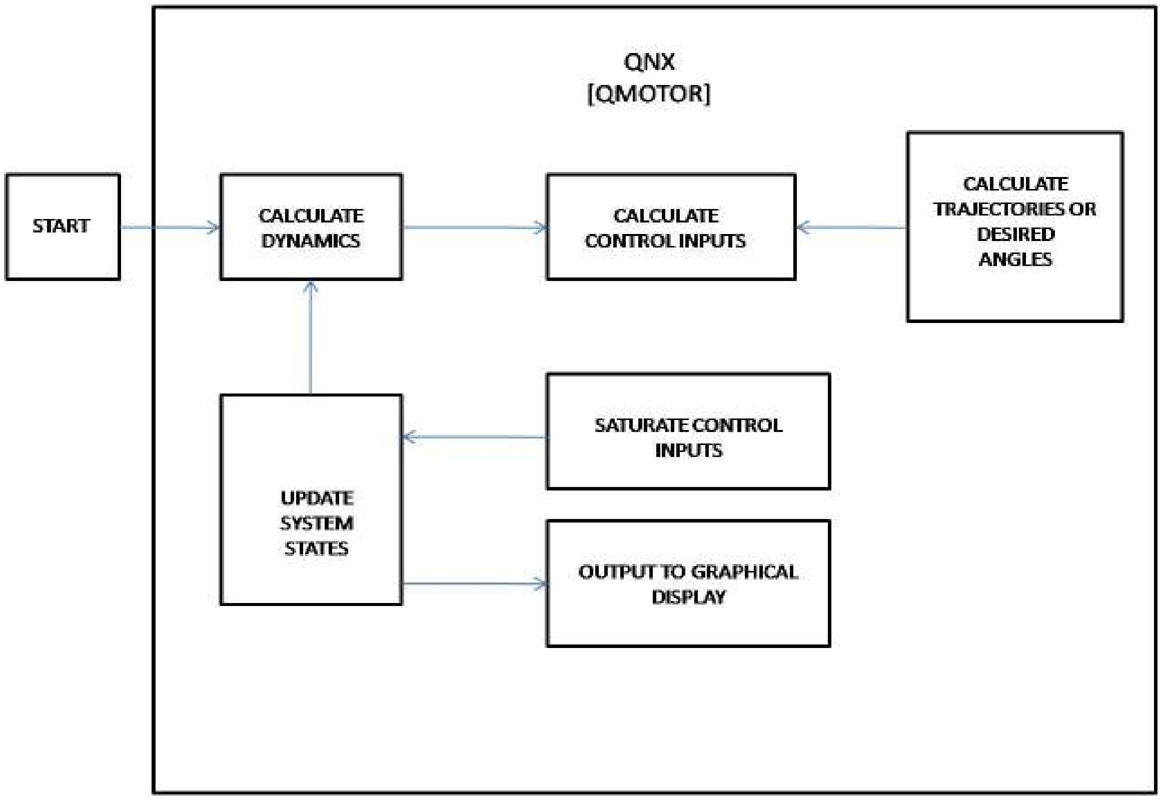


Figure 2.8 Simulation Control Structure

Simulations were done using the values formulated using the dynamic equations (2.3) - (2.5) and (2.11). These values are re-written as

$$\dot{x}_{IF}^{I} = R_F^I v_{IF}^F \tag{2.23}$$

$$\dot{v}_{IF}^{F} = -S(w_{IF}^F)v_{IF}^F + gR_I^F e_3 + F_f^F \tag{2.24}$$

$$\dot{\Theta}_{IF}^{F} = J^{-1} w_{IF}^F \tag{2.25}$$

$$\dot{w}_{IF}^{F} = M^{-1}(-S(w_{IF}^F)J_F w_{IF}^F + F_t^F) \tag{2.26}$$

Equation (2.11) remains unchanged and (2.3) is divided by m to get $\dot{v}_{IF}^{F}(t)$. Equations (2.4) is replaced by angles in (2.25) using the Jacobian (2.10). The equations are integrated using an Adams Integrator at both sides using a frequency of 1000 Hz.

The simulation is run on a QNX Real Time Operating System [26] running a QMotor [27] program using C++. The program consists of seven parts as outlined in Figure 2.8. The model is reset in start, initializing the variables. In "Calculate Dynamics", equations (2.23) – (2.26) are used to find the actual position, velocity, angular velocities and orientation of the quadrotor. In the "Calculate Control Inputs", the program uses the either the trajectory generated or the desired angles, depending on which simulation is run, to find the control inputs $[u_1\ u_2\ u_3\ u_4]$. "Saturate Control Inputs" makes sure that the inputs do not exceed the bounds of the actual system and in such case assigns them the maximum value possible from Table 1.1. "Update System States" is where all the position and velocities for the UAV and inertial frame are calculated for use in calculations in the next control cycle. "Output to Graphical Display" displays the current and desired positions, velocities, angular velocities and angles for checking purposes.

## Trajectory Generation for Position

For the position controller, the desired position is to be achieved in steps by giving a commanded position which the quadrotor follows with an actual position. Directly using the error difference between the desired and actual position will cause the creation of a large control input which thus will cause the control inputs to be running at full force and torque at all times. Not only is it not safe for the "health" of the quadrotor, but also cause the quadrotor to destabilize due to sudden maximum thrust and/or angular torques.

The trajectory generator function uses a ramp function to generate the trajectory of the quadrotor. The quadrotor position at start of time of trajectory generation is given by $p_1^I$ where

$$p_1^I = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}^I \tag{2.27}$$

and the desired position to be achieved by the quadrotor is given by $p_2^I$ where

$$p_2^I = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}^I \tag{2.28}$$

Using these desired and actual position values, an error position $p_e^I \in \mathbb{R}^3$ can be computed using

$$p_e^I = p_2^I - p_1^I \tag{2.29}$$

Using this error $p_e^I$, we can calculate the commanded position $p_c^I \in \mathbb{R}^3$ based on the following equation

$$p_c^I = (p_e^I \times ramp(t) / DT) + p_1^I \tag{2.30}$$

where $ramp(t)$ is a ramp function starting at time t (Figure 2.9) and $DT$ is a constant which dictates how fast the commanded position reaches the desired position.

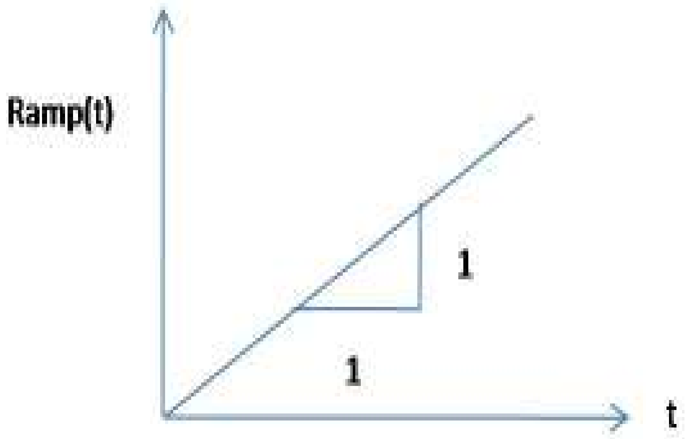


Figure 2.9 Ramp Function with a 1 to 1 output

The commanded positions are saturated and bounded between the desired and actual velocity to ensure they do not exceed limits. These commanded positions are then used instead of the desired position command in Equation (2.13).

## Simulation Results

The simulation uses the quadrotor dynamics to estimate the states of the quadrotor and find the actual states against the desired states. The control gain parameters used would not be same as the actual experiments due to the simulation torques being given directly to the quadrotor as input while in the actual case voltage is used to control the quadrotor.

For the attitude control simulation, the simulator was given a sine wave for all orientations as desired angle for 20 seconds and then immediately given an input of zero radians as desired angles for the next 10 seconds for stable hover check. The amplitude of the sine wave is 0.2 radians (11.46 degrees) and with a frequency of 0.1 Hz. The reason for such a desired signal is to see if the quadrotor can settle into a hover flight condition, as desired position eventually requires the quadrotor to reach the position and to maintain the orientation.

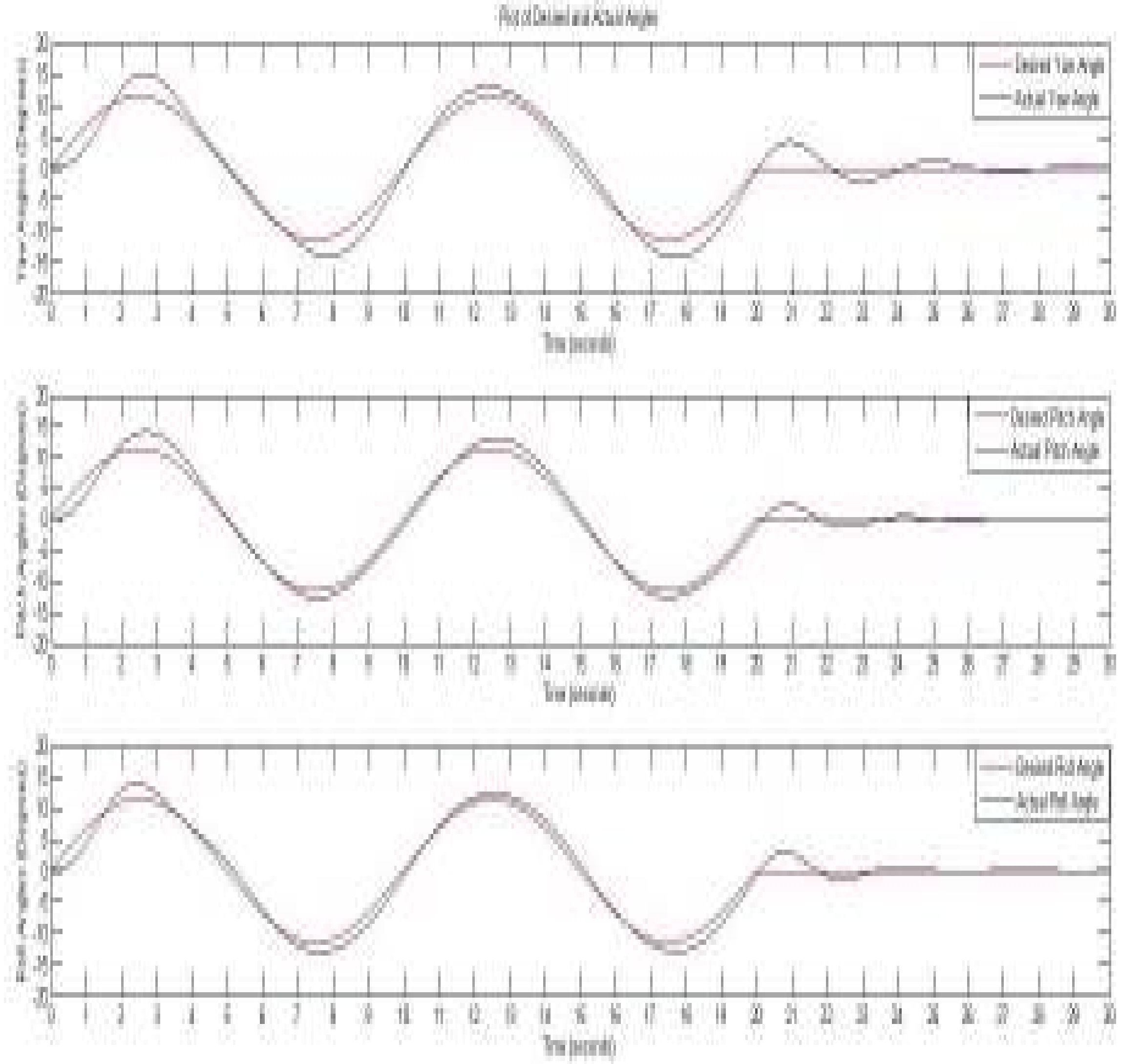


Figure 2.10. Desired and simulated actual orientations of the quadrotor

The desired angles with the corresponding actual angles are shown in Figure 2.10. The actual angles follow the desired angles closely. Yaw has a maximum variance of ±0.065 radians (3.74 degrees) from the desired angle. Roll and pitch have a maximum variance of ±0.044 radians (2.55 degrees) from the desired angle. These error peaks are observed at sudden change in direction of the desired angles.

After 20 seconds, the desired angle is zero radians. The actual angles have a maximum variance of ± 0.07 radians (4 degrees) and then settle in a damped sinusoidal pattern to zero. In this simulation, gain parameters are set as $k_{\Theta p} = [5\ 5\ 5]^T$, $k_{\Theta i} = [0.2\ 0.2\ 0.2]^T$ and $k_{\Theta d} = [0.75\ 0.75\ 0.75]^T$.

The position controller simulation was performed by using the trajectory generation function described in the section “Trajectory Generation for Position” in this chapter. The quadrotor is started initially at a reference point of $[0,0,0]^T$ and has to achieve a desired position of $[10,10,-10]^T$ with respect to the initial position. This means the quadrotor needs to move north by 10 meters, east by 10 meters and go down by -10 meters according to the NED frame used here. Going down by -10 meters is the same as going up by 10 meters.

After 20 seconds, which is the time given for it reach the position and hover, the quadrotor is commanded to come back from its new position back to its old position, meaning it has to move from $[10\ 10\ -10]^T$ to $[0\ 0\ 0]^T$ in NED frame using the same trajectory generation function used before.

The results of the desired, commanded and actual positions are shown in Figure 2.11. The results of the resulting desired and actual velocities are shown in Figure 2.12 while the results of the desired and actual orientation values are shown in Figure 2.13.

The Figure 2.11 showing the desired, commanded and actual positions shows the desired position in red, commanded position in blue and the actual simulated position in green. As can be seen from the graphs, the actual position follows the commanded position with a small overshoot when it reaches its target and then settling down to hover at that position. The north and east position have an overshoot of 0.7 meters each when the quadrotor reaches its desired position, which it does in 10 seconds time. The remaining 10 seconds, it removes the steady state error before the second part of the control program is run. The quadrotor follows the commanded position in the down direction satisfactorily, closely keeping track with it. It reaches its target of -10 meters down (10 meters up) in 5 seconds and then continues to hover at that position for the next 15 seconds.

At 20 seconds past the start of the control program, a new desired position is given to the simulation. From the acquired position of $[10\,10\,-10]^T$ , the quadrotor is commanded to go to its initial position of $[0\;0\;0]^T$. The trajectory generator again plots a commanded position, which the quadrotor has to follow. Figure 2.11, 20 seconds past the start, shows the desired, commanded and actual position.

North and East positions follow the commanded position back to the desired position, with the simulated quadrotor taking 10 seconds to reach its target. There is an overshoot of 0.64 meters which is corrected by the controller in the next 7 seconds. The quadrotor down position again follows the commanded position well, and reaches the desired position in 5 seconds, after which it continues to hover at that location.

Figure 2.12 shows the graphs for the desired velocities and the actual simulated quadrotor velocities. The quadrotor velocity graph is damped for the north and east directions compared to the desired velocities and follow the desired velocity at lower speeds. However for the down direction, the quadrotor velocity follows the desired velocity very closely. It shows a unique graph where the quadrotor accelerates for a small amount of time and then de-accelerates in the same amount of time to attain its desired velocity. With this small acceleration time, the inertia of the quadrotor is being allowed to take care of the rest of the motion.

The quadrotor attains a maximum velocity of 1.69 meters/second in the north and east directions while it attains a maximum absolute value of 3.4 meters/second in the down direction. Between ascending and descending, there is slight difference between the velocities attained, which is attributed to gravity pulling the quadrotor down while descending.

The gain values for the position simulation are set as $k_{pp} = [1\,1\,3]$, $k_{pi} = [\ 0.004\ 0.004\ 0.01]$, $k_{pd} = [\ 0.8\ 0.8\ 0.5]$. The gain values for the velocity part are set as $k_{vp} = [\ 0.0265\ 0.0265\ 8]$, $k_{vi} = [\ 0.001\ 0.001\ 0.01]$ and $k_{vd} = [\ 0.005\ 0.005\ 0.01]$. The gain values used for the orientation part are set as $k_{\Theta p} = [\ 5\ 5\ 5]$, $k_{\Theta i} = [\ 0.2\ 0.2\ 0.2]$ and $k_{\Theta d} = [\ 0.75\ 0.75\ 0.75]$.

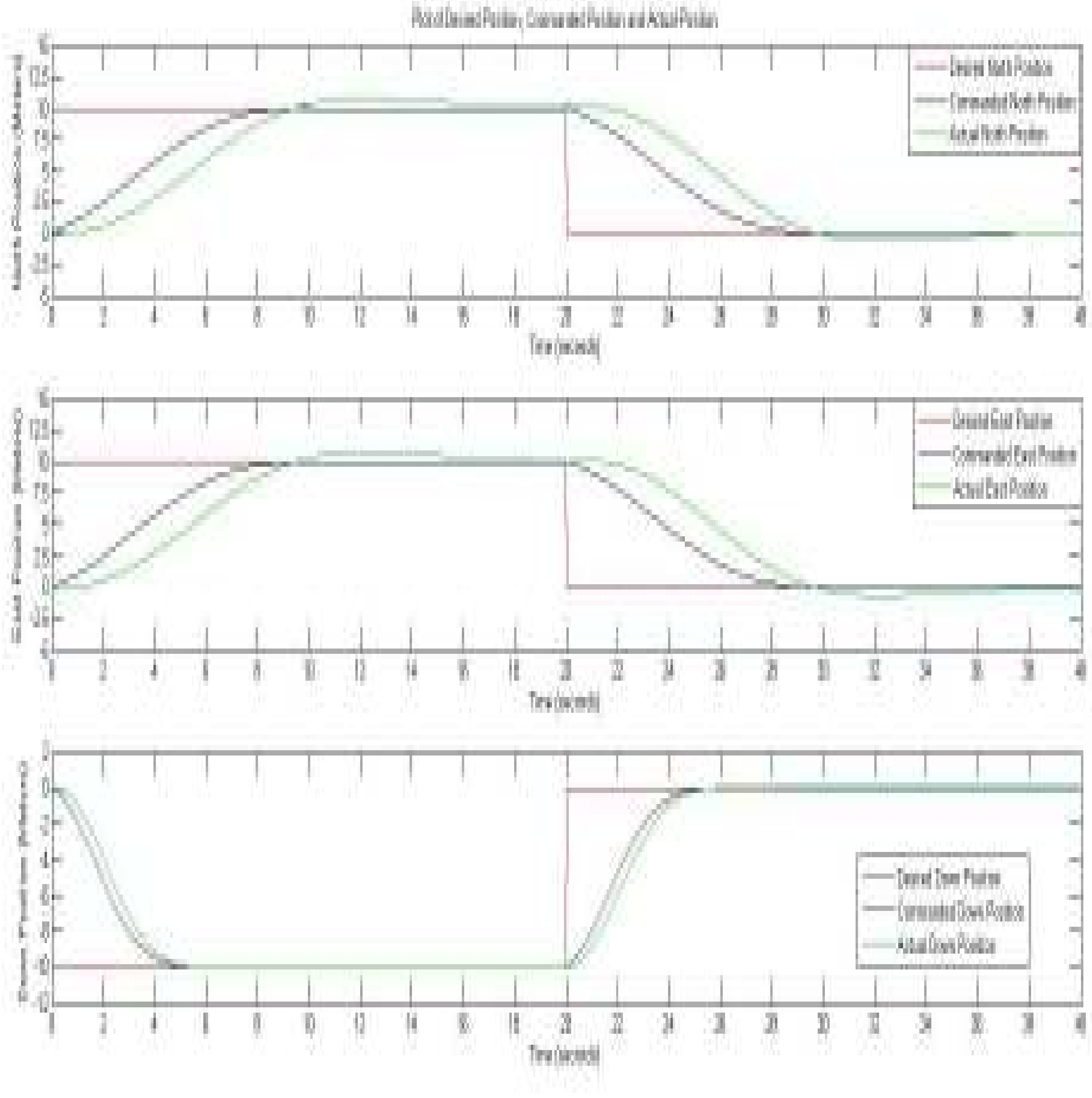


Figure 2.11 Desired, commanded and actual simulated position

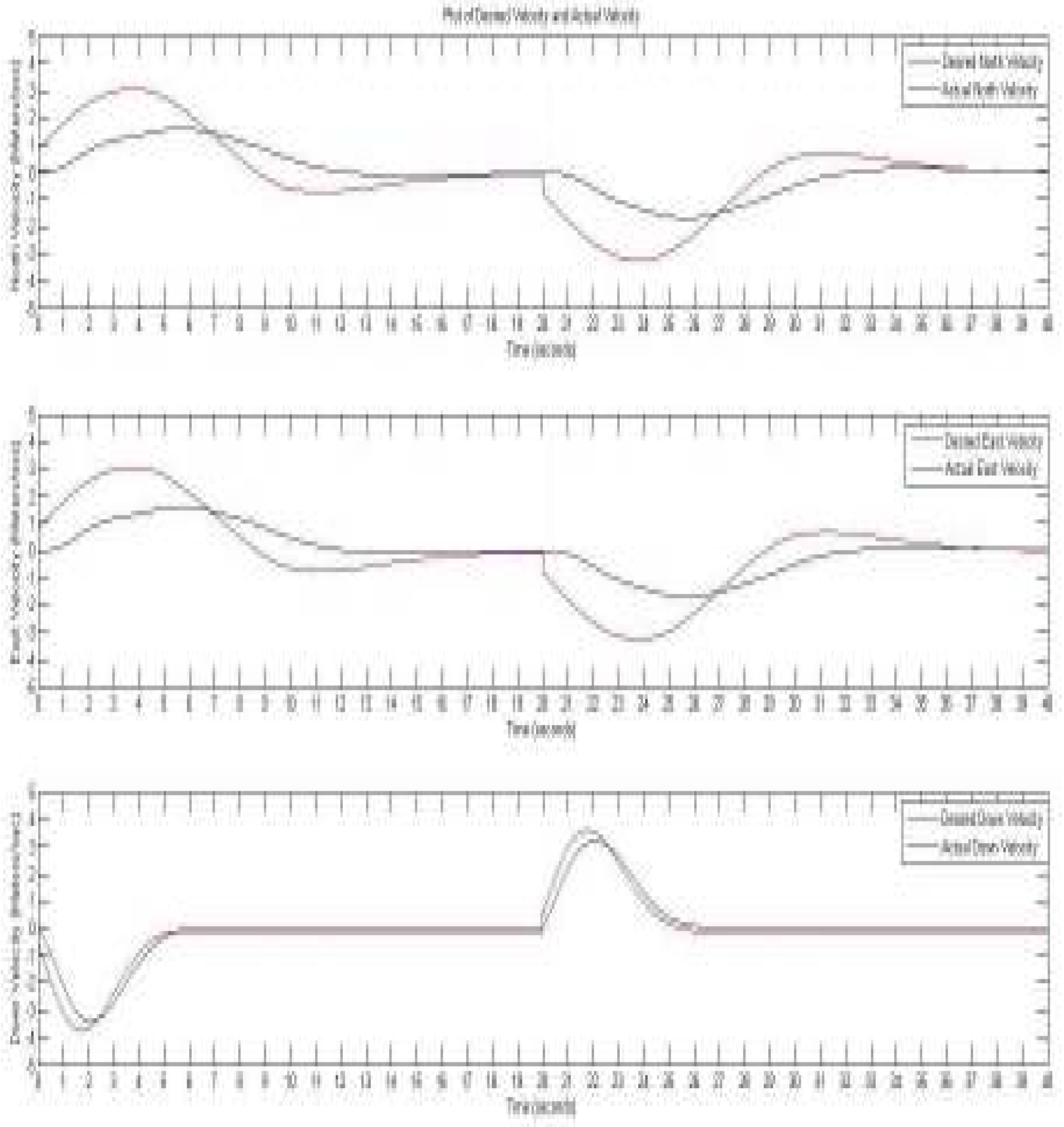


Figure 2.12 Desired and Actual Velocities

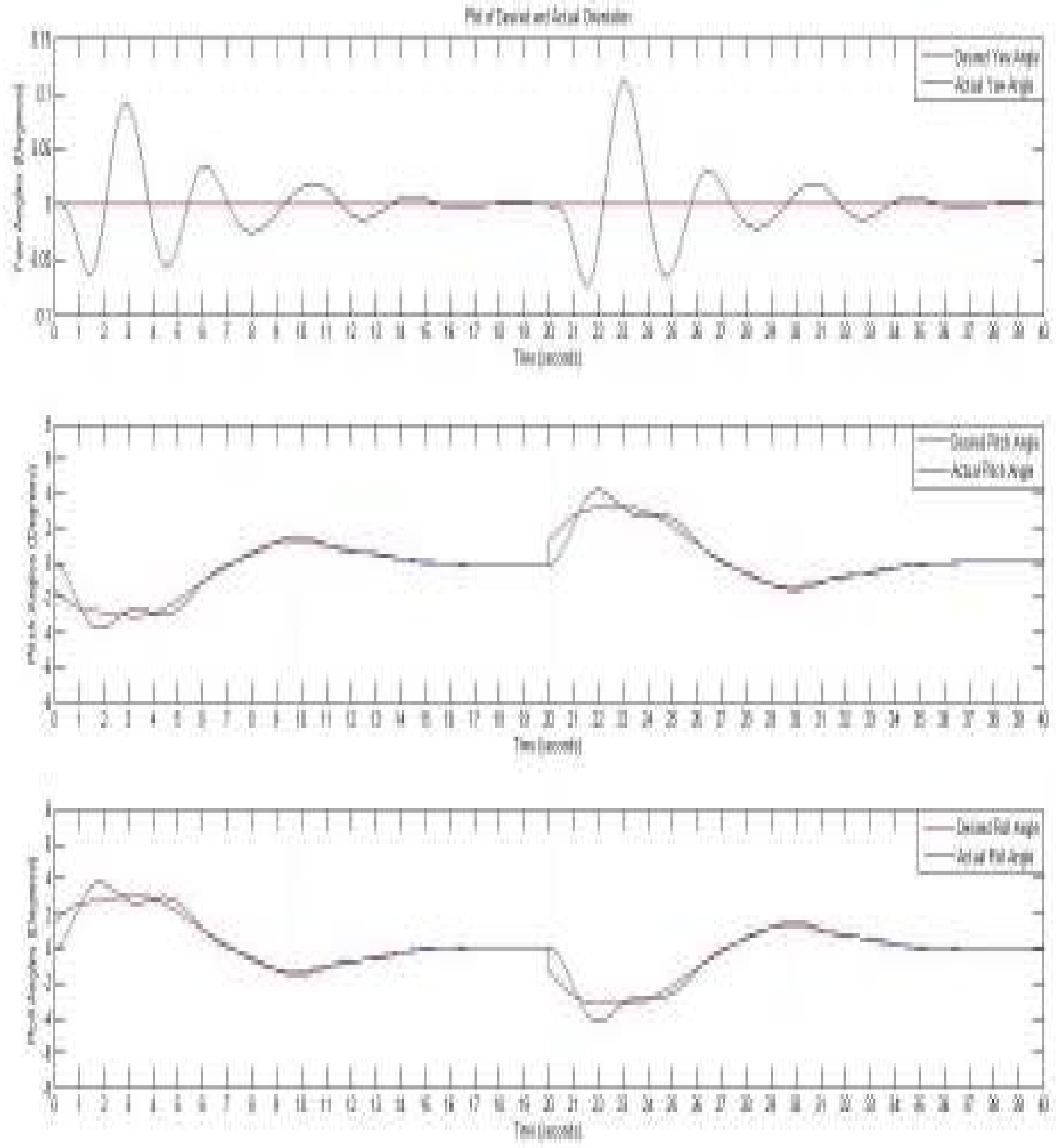


Figure 2.13 Desired and Actual Orientations

The desired and actual orientation is the key feature of this controller as it eventually provides the position of the quadrotor. Figure 2.13 contains the graphs of desired orientation with the actual orientation. The desired orientation is closely followed by the actual orientations. Yaw has a maximum variance of 0.11 degrees, and it settles to zero in a damped sinusoidal pattern. Roll and pitch angles closely follow the desired orientation, keeping a maximum variance of 1.5 degrees. The overall results show the controller working towards the quadrotor attaining its desired position and orientation with desired velocities computed being tracked as well.

# CHAPTER THREE

# EXPERIMENTAL SETUP AND RESULTS

## Introduction

For the actual experiments, a similar approach to the simulations was taken. The difference between the simulations and the actual experiments is that instead of the "Calculate Dynamics" part in Figure 2.8, there will be a "Read Sensors" part. Instead of calculating the dynamics of the quadrotor to find the states, an IMU (Inertial Measurement Unit) is used to provide the position, velocity and orientation data back to the controller.

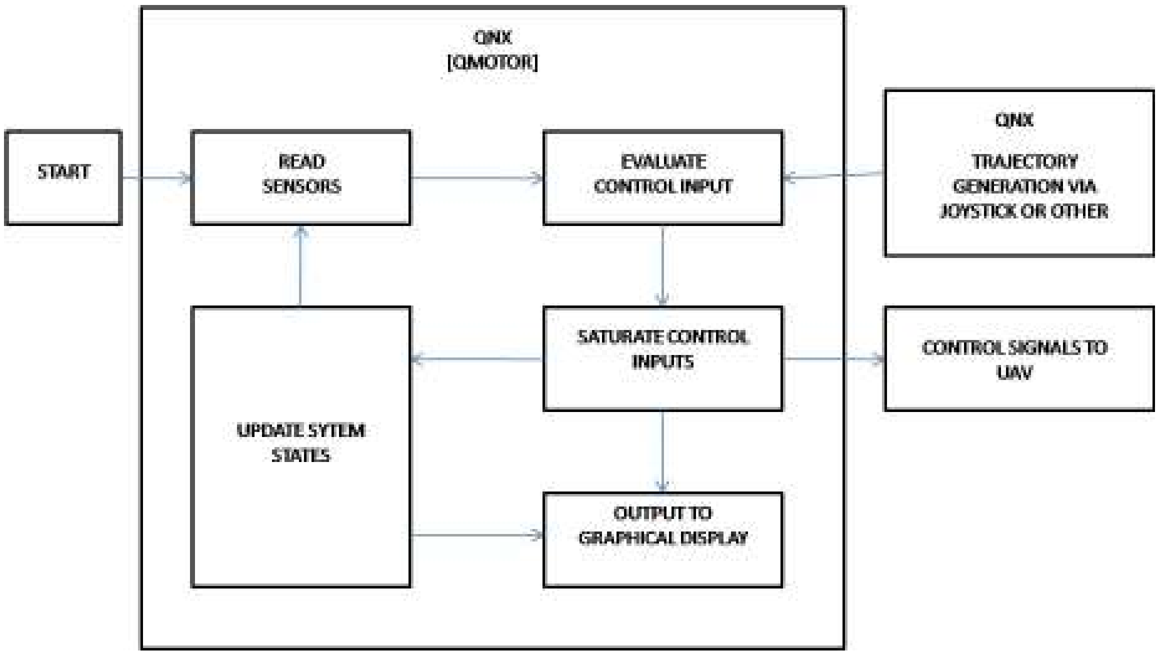


Figure 3.1 Experiment Control Structure

Another difference is that the control signals calculated by the controller are sent to the UAV through the “Control Signals to UAV” section as shown in Figure 3.1. This is done by sending the control commands directly to the DraganFlyer transceiver, via an R/C remote controller.

This chapter is divided into five sections, including the introduction part. The second section identifies the sensors and equipments used in the experiment. In the experimental setup, the choice of the sensor and the feedback loop system are explained. It then describes how the controller was trained. The third section shows the results of the attitude control experiments. We were unable to perform the position controller experiments till the point of the thesis due to large GPS error (Appendix), problems faced with converting the calculated thrust to corresponding voltage signals to be outputted by the remote controller and varying responses of motors with increase and decrease in thrust. However experiments are still ongoing for the tests. The fourth section describes the problems faced for the position control test and possible workarounds that are being tested. The last section concludes the chapter.

## Experimental Setup

To perform the actual experiments, a setup as shown in Figure 3.1 has to be made. The DraganFlyer has to be equipped with transmitters and receivers for communication for sending and receiving data. Also, the control signals calculated off board on a computer have to be converted to analog form and transmitted to the quadrotor. A sensor has to be used to find the position, velocity and orientation of the quadrotor.

The control program is run on a QNX Real Time Operating System [26] using QMotor [27] with programming being done in C++. The control program has a control frequency of 50 Hz, which means the calculations and updating are done at 50 times a second. This control program runs the controller, whether attitude or position, and can generate the desired trajectory for the attitude or position via a joystick or through the program itself.

The control signals are sent to the quadrotor using a PCM 9XII R/C remote controller shown in Figure 3.2. The computer sends the control signals to a ServoToGo breakout board (MultiQ board) to send the voltage commands to the quadrotor. The control signals for thrust, roll, pitch and yaw ($u_1(t)$ to $u_4(t)$) are sent to 4 DACs (Digital to Analog Channels) on the breakout board. These signals are within a range of 0 to 5 volts and are then directly sent to the remote controller for transmission to a receiver on the quadrotor. The remote controller works as potentiometer which uses the position of the thumb stick to send signals between 0 to 5 volts. The remote controller has been modified so that it ignores the position of the thumb stick and transmits the voltage it receives from the DACs.

For sensing purposes, a MIDG II sensor was chosen as the INS (Inertial Navigation System) of choice for the project and was mounted on the quadrotor. The MIDG II sensor has 3 axis gyroscopes, 3 axis accelerometers, 3 axis magnetometers and a Global Position system. It provides updates for the orientations at a 50 Hz frequency and GPS updates are at 5 Hz. Using these sensors, the MIDG II determines the position, velocities and orientation of the system. The sensor data output is in Microbotics Binary

Protocol. The sensor data output is transmitted via a XTend RS-232/RS-485 RF Modem [29] , shown in Figure 3.3, which also converts the MIDG RS-422 signal to RS-232 for wireless broadcast. The second X-Tend Modem uses this broadcast signal to give RS-232 signals to the QNX computer. Using software written to receive and parse the data, the measurements can be relayed to the controller. This completes the feedback loop .

## MIDG II Sensor

The MIDG II sensor includes 3 axis gyroscopes, 3 axis accelerometers, 3 axis magnetometers and a Global Positioning System. Using these sensors, the MIFG II can determine its orientation, position and velocity. It has the advantage of weighing only 55 grams as the payload of the quadrotor is limited. Using an XTend modem, sensor data can be wirelessly transmitted to the ground.

The 3 axis gyroscopes are used to find the angular rates as the MIDG II rotates about the x, y and z axes. To get the orientation, these angular rates are integrated. However there will be an initial condition problem where the values cannot be integrated, along with a drift in the angles due to imperfections in the gyroscopes.

To combat these problems, another group of measurements are required. Accelerometers measure the gravity vector and can measure pitch and roll angles. Yaw being in the gravity vector itself cannot be measured using accelerometers. To calculate yaw, the magnetometers are used to measure where north is, using north as zero degrees yaw. Using these second group of measurements, a Kalman Filter is used to determine a a bias correction for accurate determination of orientation.

However, these second group of measurements have their own failings. The accelerometers introduce additional errors as they are accelerated sideways, however this is a small error compared to acceleration by gravity. Magnetometers measure the earths magnetic field and will introduce errors if they are in the presence of any other magnetic field. Due to the high current being transmitted all over the DraganFlyer, a magnetic field is all around the magnetometers. Microbotics [31] introduced a modified firmware, which enables the MIDG to use the magnetometers to determine the initial bias for yaw, then stop using the magnetometers and allow for a small drift over time, covered in Appendix.

The MIDG II sensor uses a ANT-GPS-UC-SMA GPS antenna for acquiring the position and velocity information. A number of experiments were performed using the GPS sensor for position and velocity, which discouraged its usage. These experiments are covered in Appendix. There were large errors in both position and velocity along with problems faced with number of updates being received by it. The GPS is supposed to update with a frequency of 5 Hz. However, sometimes updates took a number of seconds.

## Quadrotor Training

For computing the desired PID values of the gains, the quadrotor needed to be checked with those values. The quadrotor was hung from a steel beam on the ceiling of the laboratory with a piece of flexible rope. The rope end had a normal hook to which a sailing hook was attached which allowed the quadrotor to yaw without problems. The quadrotor was initially set up with loose ropes attached to the arms of the quadrotor to the arms of a stand right below it as shown in Figure 3.4. This prevented the quadrotor from yawing, pitching or rolling dangerously. Once it was certain that the quadrotor had just sufficient freedom of movement, the zero values of the quadrotor were found for roll, pitch and yaw. Zero values are the voltage values sent to the quadrotor which ensure that the quadrotor does not roll, pitch or yaw without any feedback control. These are calculated by sending voltage signals to the quadrotor through the controller and checking the response of the quadrotor to those signals.

Once the zero values have been found, the quadrotor orientation values can be tuned for attitude control. The thrust for the quadrotor is given through the slider of a Wingman 3D Extreme Joystick, shown in Figure 3.4. Once the gain parameters have been tuned, the ropes on the arms of the quadrotor are removed and the quadrotor is allowed to hover in the air with the controller taking care of the orientation.

Figure 3.2 PCM 9XII RC Remote Controller

Figure 3.3 XTend RF Modem XT09-PKG

Figure 3.4 Wingman Extreme 3D Joystick

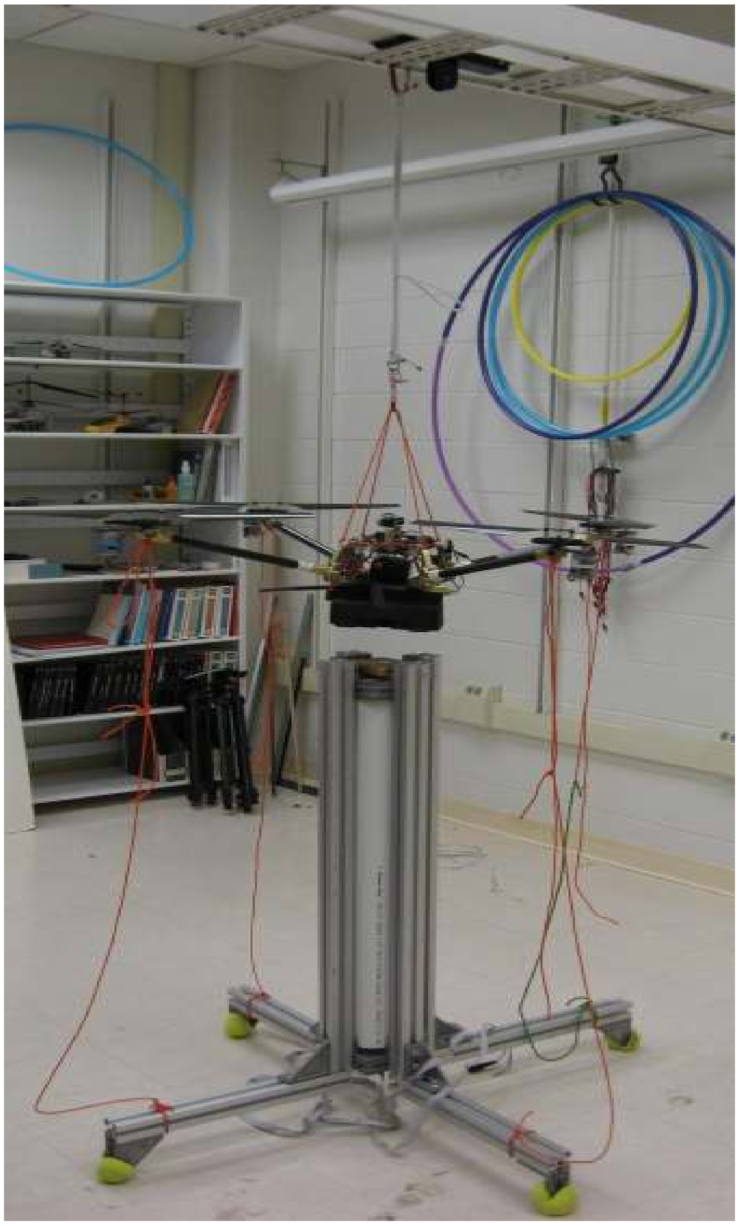


Figure 3.5 Quadrotor training stand

## Results

The attitude controller shown in Figure 3.6 was tested in a room to check whether the quadrotor can autonomously control its orientation with thrust being provided by the joystick slider. Once the quadrotor reached hover thrust, the readings of the orientation of the quadrotor were noted. Other tests were also done to check the attitude stabilization. The quadrotor was given a sine wave as desired angle input at several different frequencies to check its response.

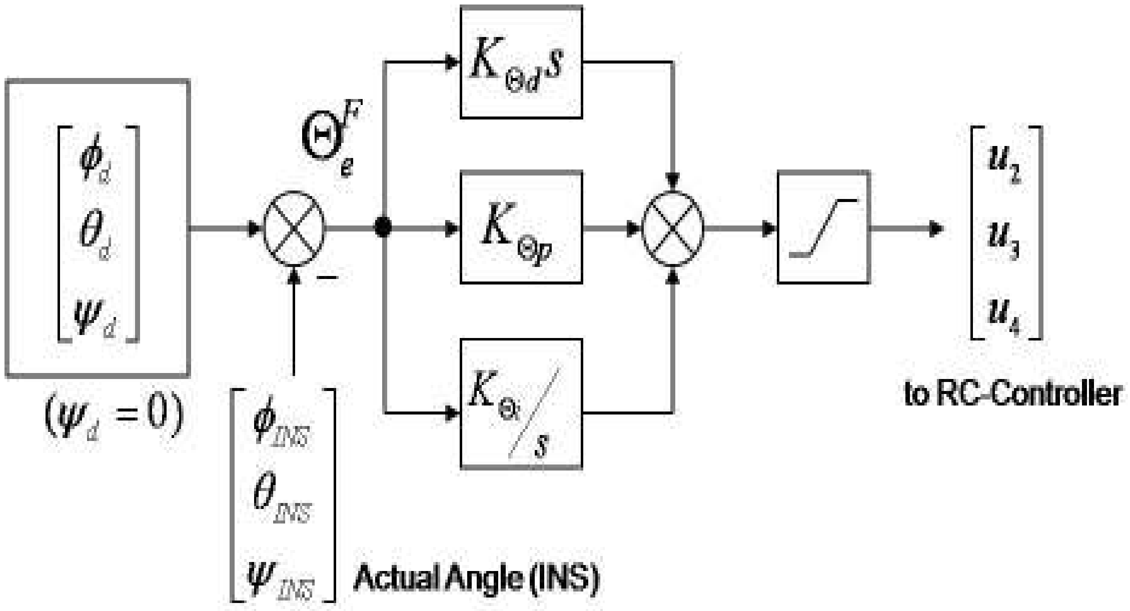


Figure 3.6 Attitude Controller

## Hover Test

The quadrotor was hung with a flexible rope and thrust was given to make it hover without using the ropes as support. The desired angles in this case are all zero, and as such the controller works to achieve it.

After preliminary testing, the quadrotor was tested with gain values of $k_{\Theta p} = [\ 4\ 4\ 4]$, $k_{\Theta i} = [\ 0.97\ 0.97\ 0.97]$ and $k_{\Theta d} = [\ 0.15\ 0.15\ 0.15]$. At these values it was observed the roll, pitch and yaw had a maximum variance of ±5 degrees, with most angles being within ±4 degree range as shown in Figure 3.7. There as a lot of position drift as there is no position control implemented, which caused the quadrotor to drift about its mean position.

After testing several gain values, the gain parameters $k_{\Theta p} = [\ 5\ 5\ 5]$, $k_{\Theta i} = [\ 0.97\ 0.97\ 0.97]$ and $k_{\Theta d} = [\ 0.11\ 0.11\ 0.11]$ were found to give the best performance for the quadrotor as shown in Figure 3.8. At these values, the quadrotor had minimal drift in position and kept the roll, pitch and yaw angle error within ±2 degrees.

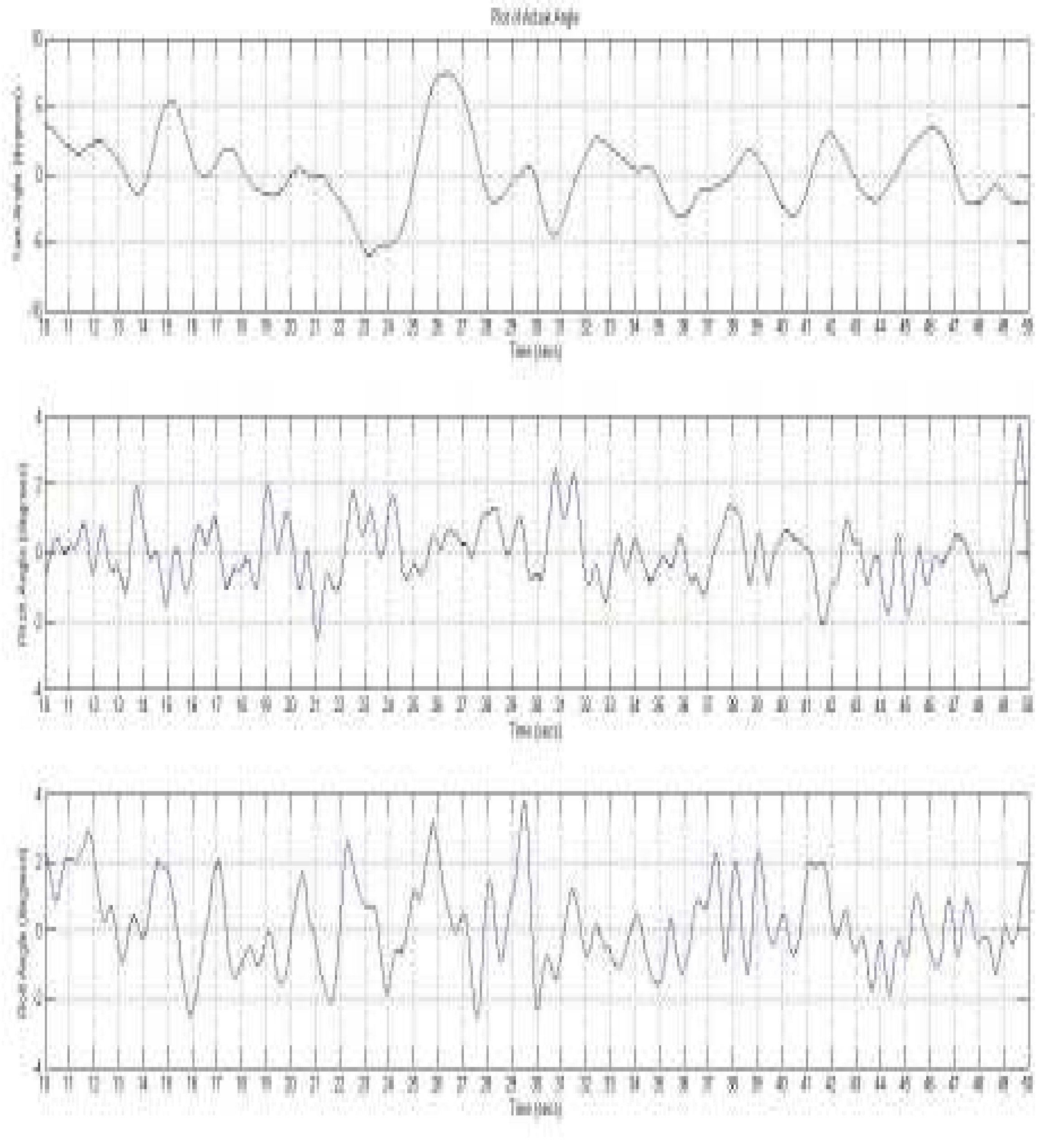


Figure 3.7 Plot of yaw, pitch and roll angles with initial gains

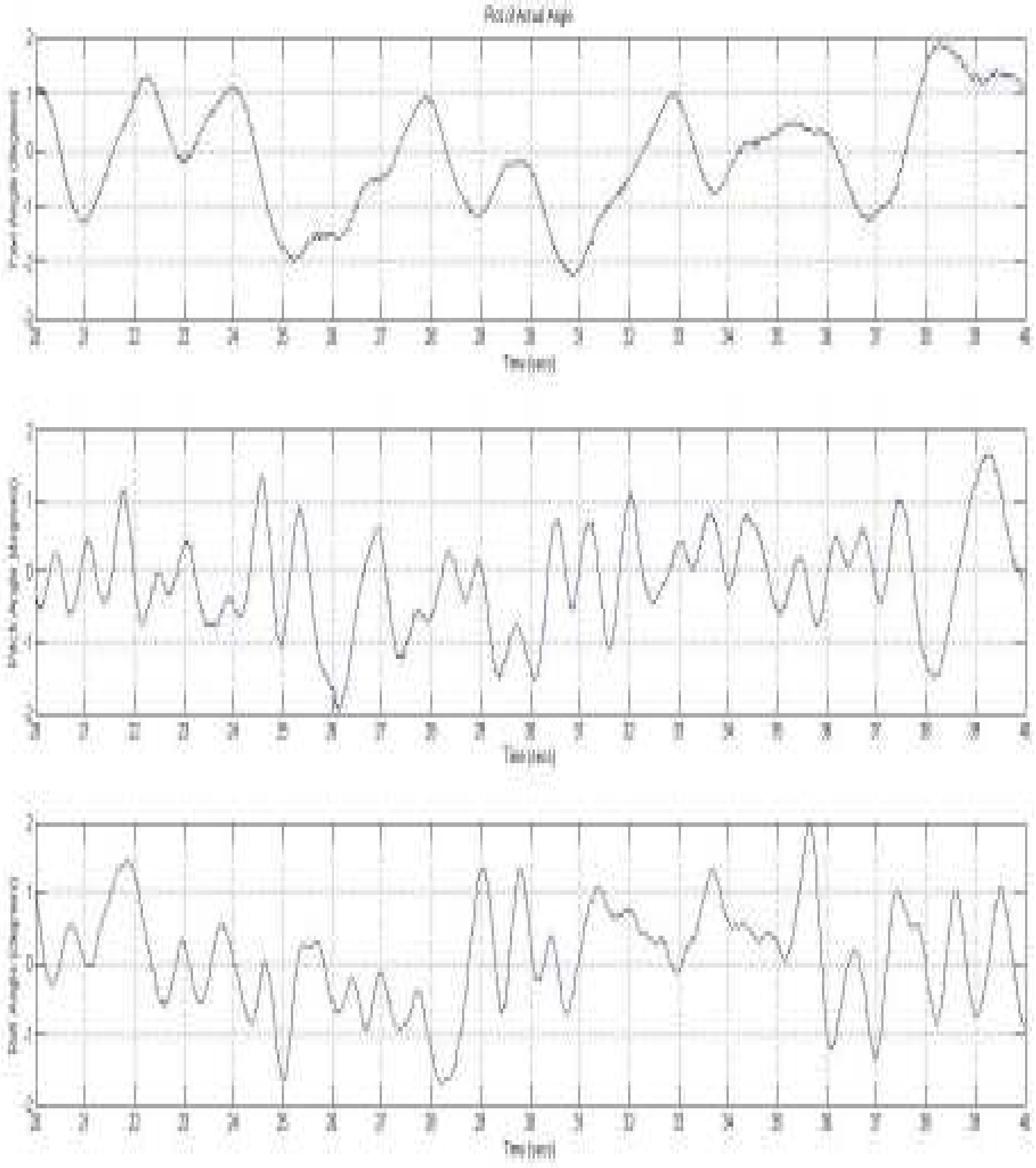


Figure 3.8 Plot of yaw, pitch and roll angles with final gains

## Sine Wave Test

Using the final PID gain parameters, desired angles were given to the quadrotor as a sine wave with amplitude of 4 degrees for roll and pitch and 7 degrees for yaw, with a frequency of 0.5, 0.75,1, 1.5 and 2 Hz. However for our purpose here, we only show the results for the 0.5 and 1 Hz frequency tests. The reason for such a test was to find the response time of the system and to check the response of the system to varying angles.

The 1 Hz test results are shown in Figure 3.9, 3.10 and 3.11 for yaw, pitch and roll respectively.

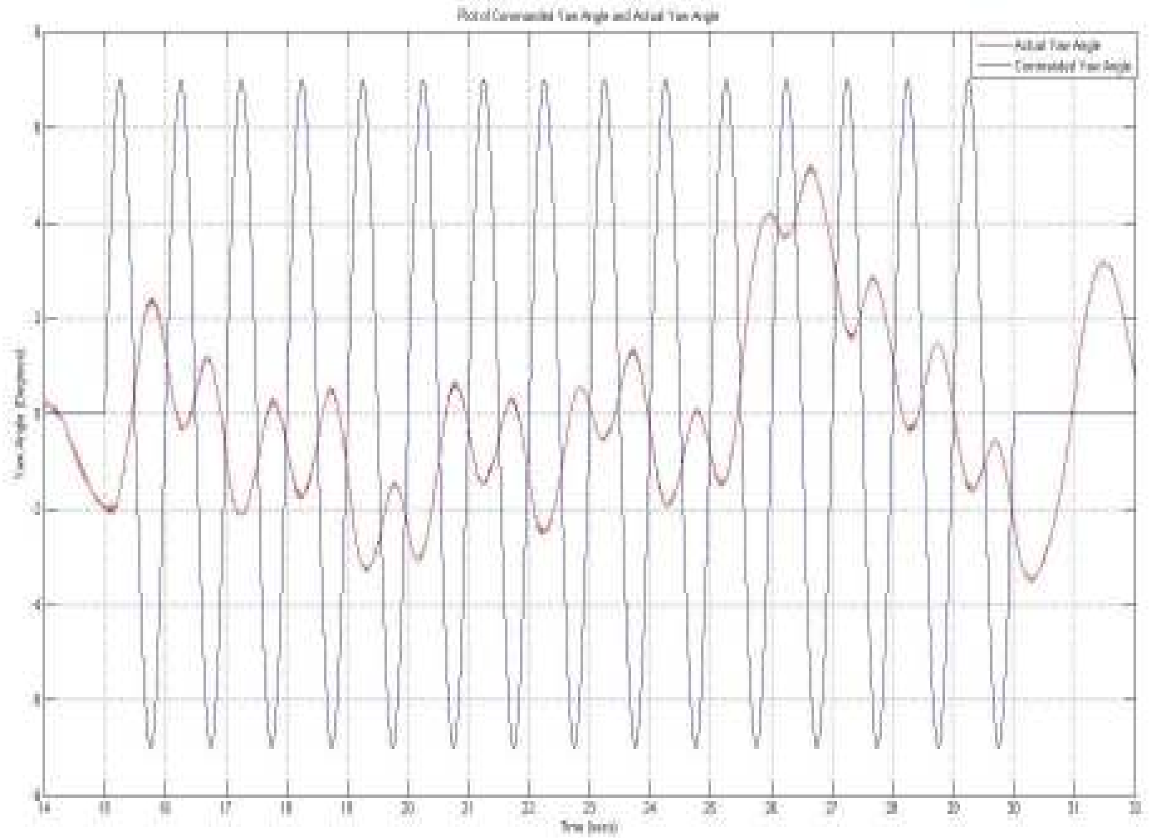


Figure 3.9 Yaw angle for 1 Hz sine wave test

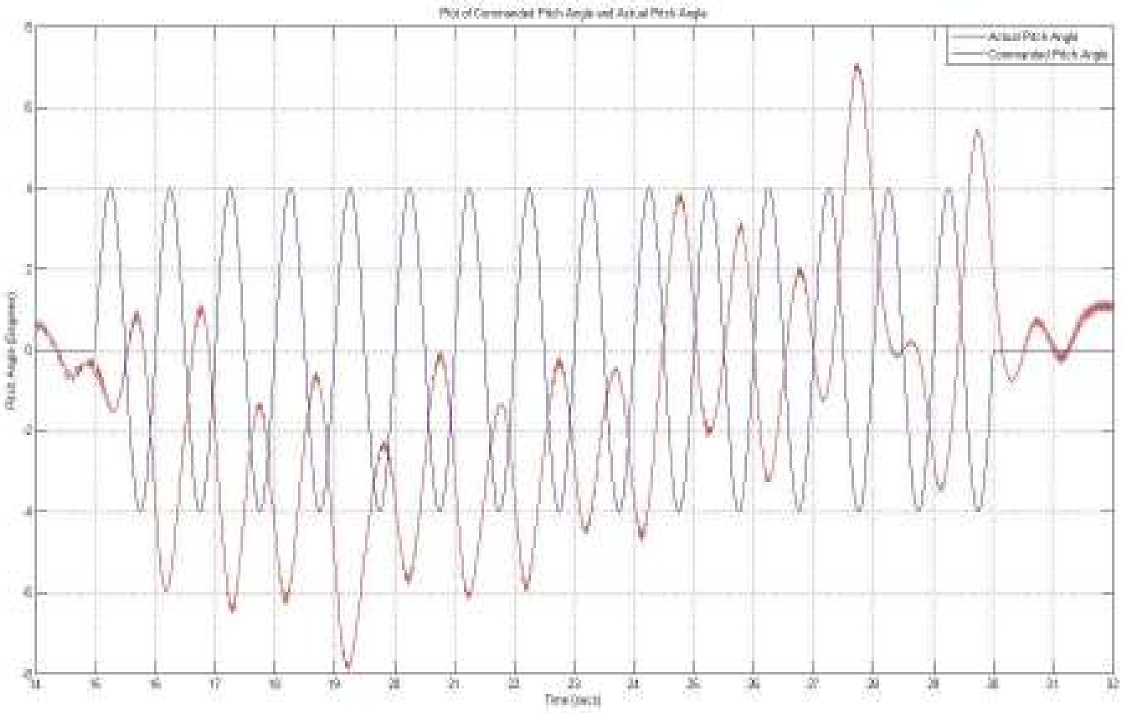


Figure 3.10 Pitch angle for 1 Hz sine wave test

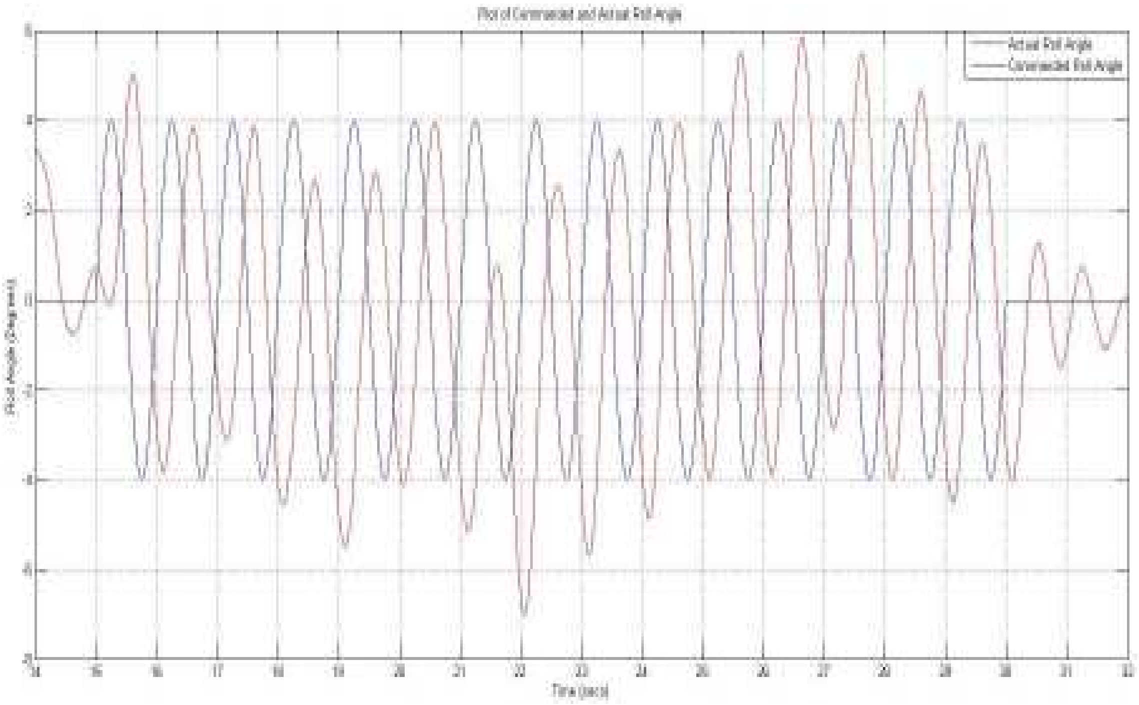


Figure 3.11 Roll Angle for 1 Hz sine wave test

As can be seen from the results, the controller tries to follow the commanded angles given to the quadrotor but due to large response time, calculated at 0.5 seconds, the responses lag is almost 180 degrees. This shows that quick adjustments in short intervals may not be suitable.

Response tests at 0.5 Hz are shown in Figures 3.12, 3.13 and 3.14 for yaw, pitch and roll angles again. Yaw angle follow the commanded angle but seems to be about 90 degrees out of phase. Roll and pitch angles however respond well to the desired angles. The response time for the system is calculated to be 0.4 seconds.

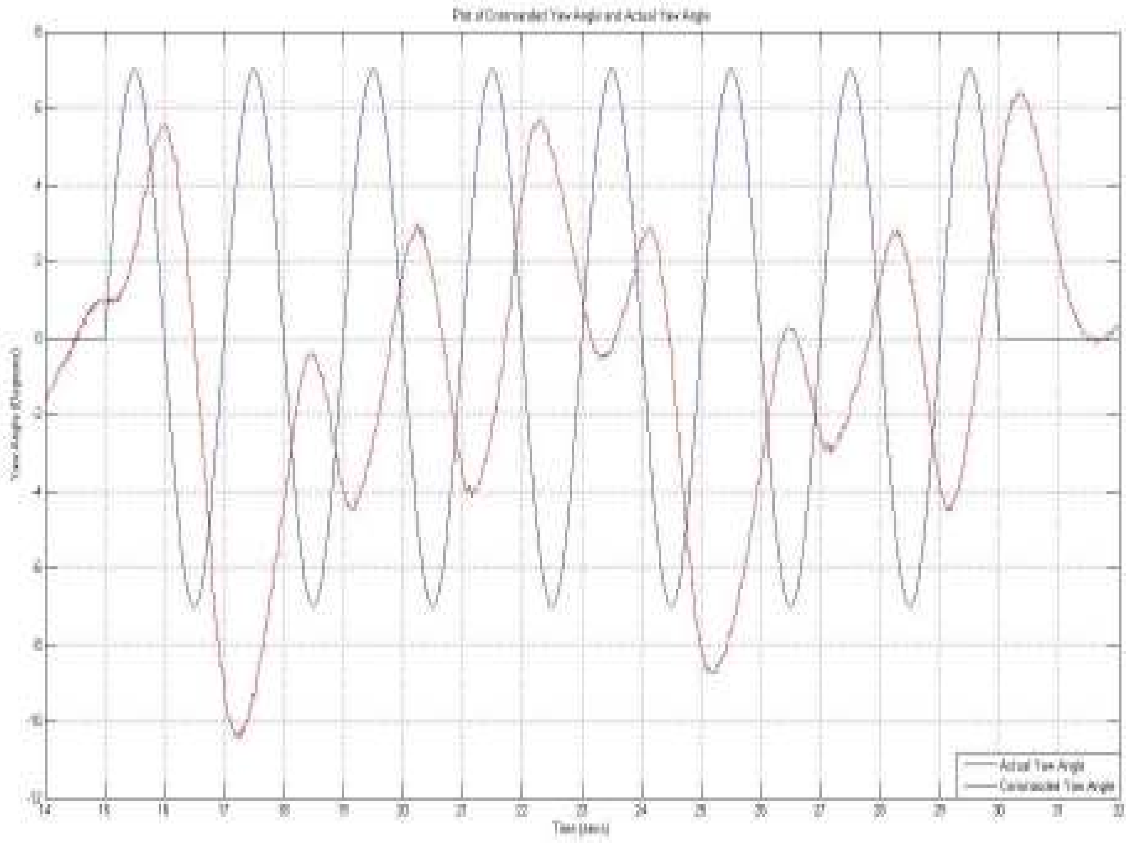


Figure 3.12 Yaw angle for 0.5 Hz sine wave test

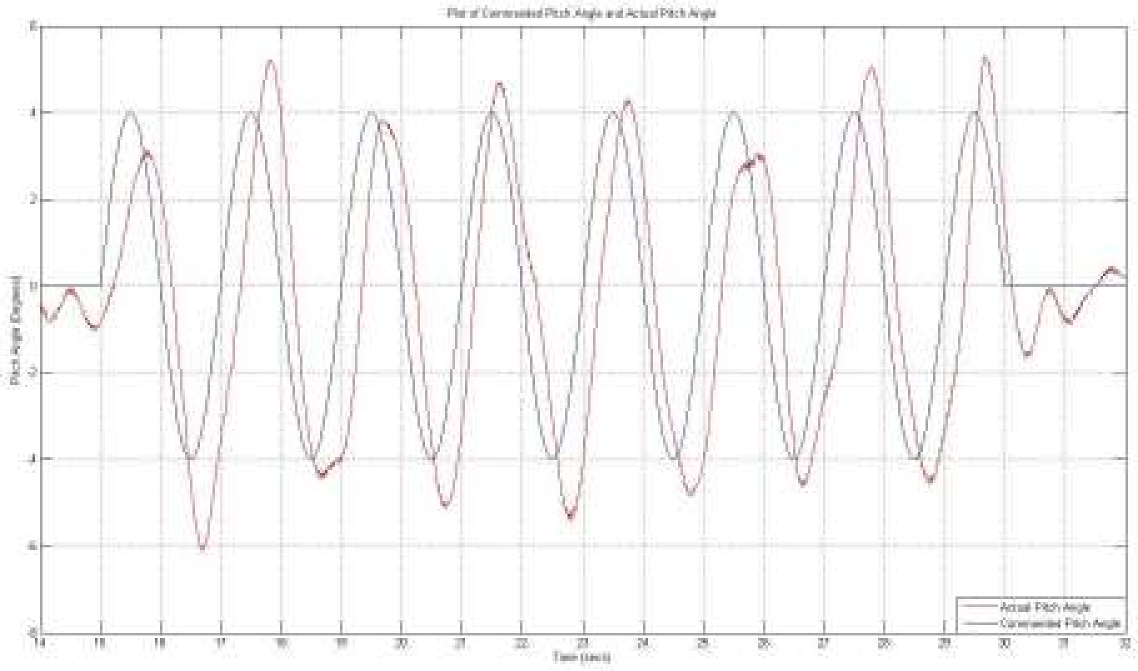


Figure 3.13 Pitch angle for 0.5 Hz sine wave test

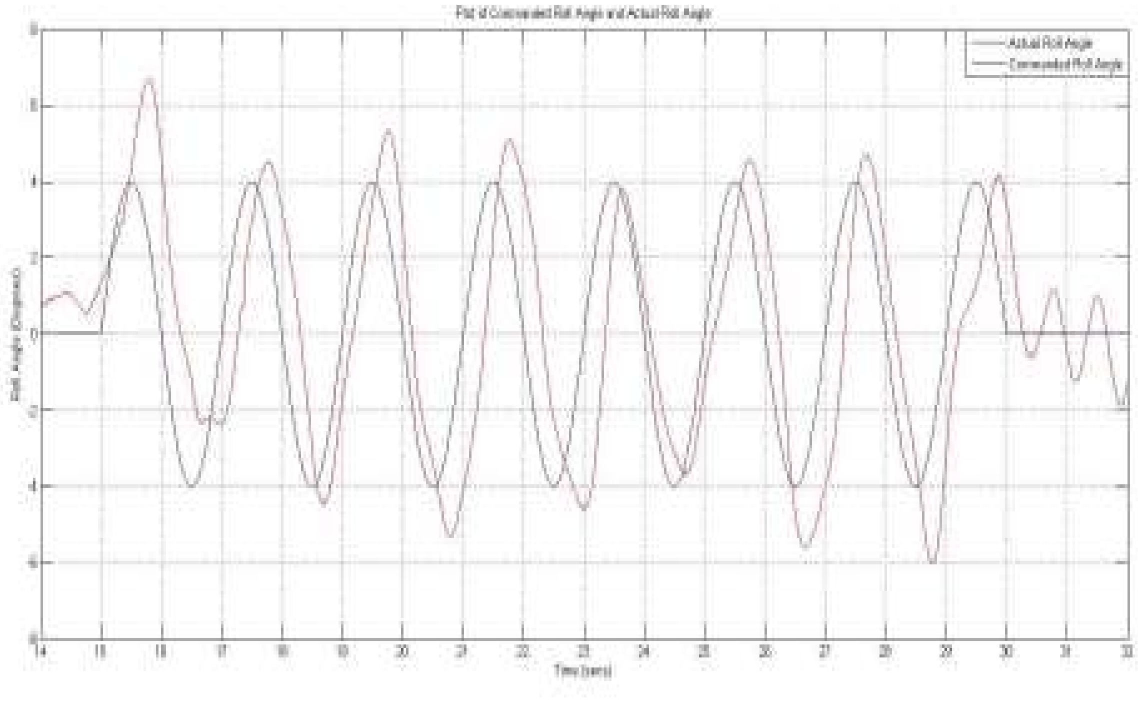


Figure 3.14 Roll Angle for 0.5 Hz sine wave test

## Position Control Problems Faced

Experimentations for the position controller are still ongoing till the point of the thesis and will be continued after. The main problems faced with implementing the position controller are outlined more expansively here. Position control needs to be ideally implemented when the quadrotor is already in the air, taking the position it is at in the air as the starting point. Trying to apply position control when the quadrotor is on the ground does not seem a very safe method from various tests performed.

When taking off from the ground, the ground effect due to the moving air, created by the rotating blades, hitting the ground and coming back up to the blades causes instability in the system and may cause the quadrotor to roll and pitch forcefully. Due to desired pitch and roll angles being given to the system from the position controller while it is on the ground, the quadrotor tends to apply an angle while it is on the ground which may topple it or cause the blades to strike the ground. The quadrotor can torque and thrust freely in the air but it cannot do so on the ground. Another issue is the large errors seen using the GPS. Near the ground with such large errors may cause the quadrotor to lose height and make it crash. Position has a maximum error of 8 meters with velocity having a maximum error of 2 meters/second in tests where the quadrotor is not moved.

For this a takeoff where only z axis and attitude are controlled may also be used. A workaround for this is to suddenly increase the throttle of the quadrotor, so that it jumps up into the air with minimal effects on the pitch and roll as the attitude controller should take care of the angles, and then settle into a position hold at a certain position using the position controller. Such a GPS hold can help us tune the gain values for the system. Position control with a desired trajectory can then be implemented through a command via a joystick or through the program. Such a method also has problems. The quadrotor can accelerate very fast and thus settling into a desired position after the jump is very difficult due to the large response time and incomplete information on which gain values to use for position and velocity.

Right now, experiments are being performed on the performing a GPS hold test on the quadrotor by hanging it from an elevated beam or branch outside and starting up the quadrotor in such a case with only attitude control implemented. Once the quadrotor attains hover, the program can be switched over to a GPS hold, using the position controller to test the gain values. These gains can be fine tuned for later testing of the quadrotor off the ground using just z axis and attitude control for takeoff and applying the trajectory once it attains a certain height.

## Conclusions

The PID controller for attitude control works well and allows the orientations to be maintained as long as thrust is given via the slider or through other means. This allows the user to fly the DraganFlyer with much more ease than previously with an open loop control. The controller maintains hover angle errors within ±2 degrees for all three axes and follows desired angle trajectories well. Maintaining the orientations allow the position loop control simulated to be implemented in ongoing experiments and future works.

# CHAPTER FOUR

# VISION SYSTEM

## Introduction

Stereo Vision is the process by which we see and estimate distances in the world around us using our two eyes. The distance between our eyes gives the brain two images of the object that we are looking at. The different images give us a slight displacement (called disparities) of the object in the two projections of the world.. The brain is then able to process these disparities further to estimate distance from the object.

Stereo vision in cameras work the very same way. There are two lenses, with the same focal length, placed some distance apart from each other, called baseline, that take two images of an object at the same time. The displacement between the camera lenses causes a displacement of the appearance of the object in the left and the right stereo images. This disparity is used to estimate distance of the object from the camera through various algorithms available.

In the case of the quadrotor experiment, stereo vision was one method that was initially thought of to be used for height estimation when close to the ground. The GPS sensors give large position errors and thus the chances of the quadrotor thinking its distance is higher or lower than the ground than it actually is arises. This may lead to crashes and may also lead to problems with takeoff and landing, if such a trajectory portion is included.

The objective of this chapter is to show that stereo vision may be utilized to estimate distances under several circumstances, using disparity between the left and right images. This chapter is divided into 4 sections with the introduction being the first section. The second section concerns the equipment used for distance estimation, with the algorithms used and how the images were captured and utilized. Different surfaces were tested to ensure that the results covered a wide range of possibilities with usage of the camera. The third section shows the results of the tests performed and discusses the accuracy and reliability of the results followed by the conclusion.

## Equipment Used And Algorithms

The camera used is a STH-DCSG-STOC stereo vision camera system designed and manufactured by Videre Design [19]. The camera has two replaceable lenses spaced apart at a fixed distance of 9 centimeters with the lenses having a focus of 4mm each. The camera has a 6 pin firewire port which connects it to a computer with an IEEE-1394 firewire 6 pin to 6 pin cable. The software used for capturing, processing and displaying the results is SVS (Small Vision System) [20], an implementation of the stereo vision algorithm developed by SRI International [31].

The stereo vision camera is an STOC (Stereo on Chip) type camera which means it has an FPGA (Field Programmable Gate Array) on it, allowing for stereo processing to be done on the camera instead of the computer. It has a global shutter so it can capture images in motion and process them. The camera uses 1.5 Watts of power for normal operation which can be provided by the batteries on board the quadrotor. It has a C++

library API for MS Windows and Linux with SVS which would allow data capture of the results and inclusion into the controller as a sensor for position estimation. The results attained from the camera can be used with the GPS data by using a Kalman Filter, for better estimation of height for low flying conditions. Providing the power to the camera can be done through the onboard batteries but transferring the data to the QNX computer for capture would entail adding another wireless or Bluetooth system on board.

For the tests, the camera is supported using a tripod to keep it steady. The tripod also prevents sudden jarring which may require the camera to be re-calibrated. The images captured by the camera are received by the SVS interface via the firewire IEEE 1394 port. The left and right images can be viewed simultaneously in the interface and real-time images can be captured continuously at 60, 30, 15, 7.5 and 3.75 frame rates per second. The size of the images can also be specified through a drop down box from 320x240 to 1024x768. A resolution of 640x480 and a frame rate of 15 fps were used for all images in the database. The left and right images can be loaded into a video buffer from the interface and then downloaded onto the computer (bmp format).

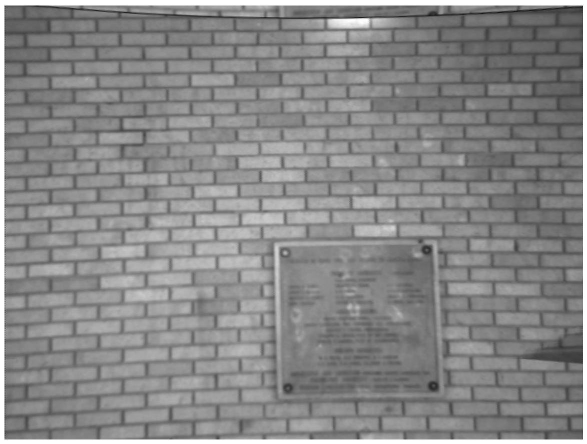


Figure 4.1 Left Image Brick Wall

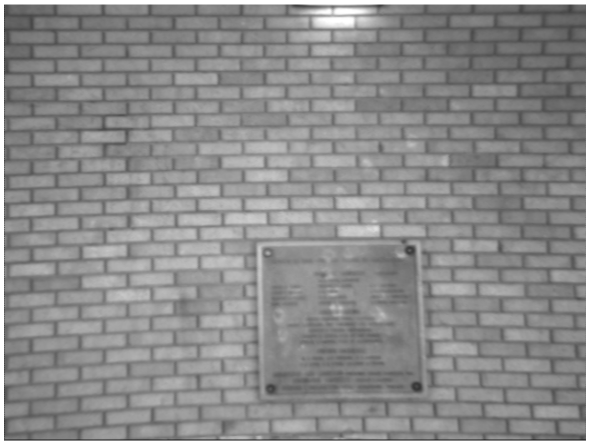


Figure 4.2 Right Image Brick Wall

For the comparison, images of different surfaces and places were acquired and the surfaces selected for image analysis were a wall, grass, shrubs, a tree trunk, pavement area and water. The reason for different surfaces is to test the reliability of the camera for distance estimation of several regions. The distance from the camera to the surface was increased in increments of 0.5 meters starting at 0.5 meters till 4 meters.

Figure 4.1 and Figure 4.2 show the left and right images of surfaces captured using the stereo vision camera using SVS at a distance of 2 meters. For calculating the distances, an area correlation algorithm is used. In the stereo vision algorithm, an area correlation algorithm is used to find disparity between the stereo images. Before correlation is done, the images are rectified and features are extracted. Rectification is done to remove any noise in the image. Calibration removes any lens distortion and take care that the vertical disparity is zero i.e. epipolar lines are aligned. Features are extracted by taking the Laplacian of Gaussian of the images. Pixels in the left image are found in the right image using a search window of 64 pixels in this case. Filtering is done to remove bad matches and then the disparities are converted to 3D points.

The calibration routine [32] [33] followed for the camera was that for a STOC camera using the SVS. Whenever the camera was moved from one place to the other for capturing images, the camera was re-calibrated to ensure correct results. The calibration routine involved a 9 x 7 squares checkerboard with a square side size of 116 mm. At least 9 images were captured of the checkerboard in different orientations for each calibration. The calibration routine provided in the SVS calibrates the images and gives the error readings: the average bias from the epipolar line, RMS error or the average deviation of the features from the ideal epipolar placement and the standard deviation of the epipolar error. These errors need to be typically within a certain range. The average bias should typically be less than 0.05 pixels and the RMS error should be within 0.1 and 0.15 pixels. These error ranges were adhered to for the experiments.

## Results And Discussions

Images were from the image set for analysis as only a pair of left and right images are needed for the distance estimation. For all surfaces and corresponding distances, the 30th image from the buffer files was picked except for water where the 55th image was picked.

With SVS, stereo analysis can be done in real time as the images are being fed into the interface. The interface can be set such that the left window shows the image from the left lens and the right window shows the disparity map with the feature points being indicated in color. As the distance of the objects from the camera increases, the color of the feature points in the disparity map change from red towards violet.

To estimate the values of the distances through the SVS disparity map, there is an option to download a 3D point array in text format. This array contains all the x, y and z distance estimate. Since the disparity bmp images are also 640x480 resolution, the numbers of pixel that give data are equal to 307,200. For a given surface, using MATLAB, data points were sorted through and only those distance estimates were used which were within the required range of co-ordinates specified. An average of all the distances gave us the final estimate of the distance.

Table 4.1 Results of stereo vision experiments

| ORIGINAL DISTANCE | ESTIMATED DISTANCE (METERS) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Wall | Water | Pavement | Grass | Checkerboard | Trunk | Shrubs |
| 0.5 m | 4.1035 | 1.1394 | 0 | 0 | 0 | 0 | 1.0348 |
| 1.0 m | 0.9504 | 1.3704 | 0.9564 | 0.9662 | 0.9701 | 0.9211 | 1.0076 |
| 1.5 m | 1.4073 | 1.6744 | 1.4219 | 1.4466 | 1.3853 | 1.4014 | 1.4411 |
| 2.0 m | 1.8992 | 2.1228 | 1.8612 | 1.9872 | 1.9028 | 1.8715 | 1.9673 |
| 2.5 m | 2.4341 | 2.481 | 2.5392 | 2.2905 | 2.3829 | 2.3787 | 2.5339 |
| 3.0 m | 2.9281 | 2.891 | 2.9269 | 2.8651 | 2.9296 | 2.9463 | 3.0528 |
| 3.5 m | 3.3919 | 3.2778 | 3.2276 | 3.3653 | 3.4507 | 3.4261 | 3.5783 |
| 4 m | 3.9064 | 3.3799 | 3.8566 | 3.9305 | 3.9452 | 3.95 | 4.0438 |

The results of the experiment are shown in Table 5.1 with all distances in meters. The first thing that stands out is that for the software, there are almost no estimates for 0.5 meters. The ones that are available seem to be false readings with only a few feature points available out of a wide possible range.

Figure 4.3 shows the graph between average error and actual distance for all surfaces. The distance estimation has a maximum error of 0.141 meters across all distances except 0.5 meters. At 0.5 meters, the false readings due to wall, water and shrub surface contribute to a higher error. Typically the error is bounded within 0.1 meters.

The results may further be classified into regular and irregular surfaces. Regular surfaces include wall, pavement, water and checker board surfaces. Irregular surfaces include grass, tree trunk and shrubs. Figure 4.4 shows the graph between error and actual distance for regular surfaces while figure 4.5 shows the graph between error and actual distance between irregular surfaces. The graph for regular surfaces shows a large error for 0.5 meters and comparatively small errors for all distances. The error for regular surfaces is typically within 0.22 meters and do not do so well compared to irregular surfaces. That is due to the inlcusion of water and pavement data which did not show as much accuracy as the others. The irregular surfaces graph has a large initial error , nevertheless it shows a maximum error of 0.1 meters after that.

## Conclusions

The stereo vision system is quite robust for different distances and errors tend to stay within a small range. It has problems with distances around 0.5 meters; however for

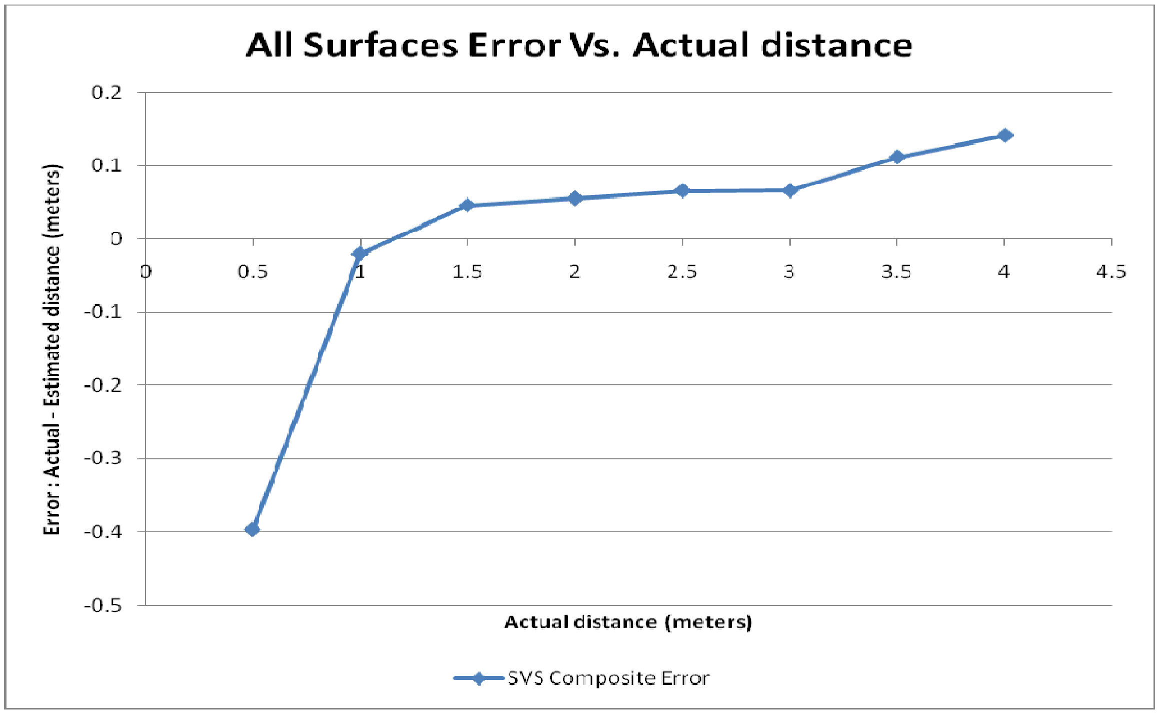


Figure 4.3 Graph for error for all surfaces vs actual distance

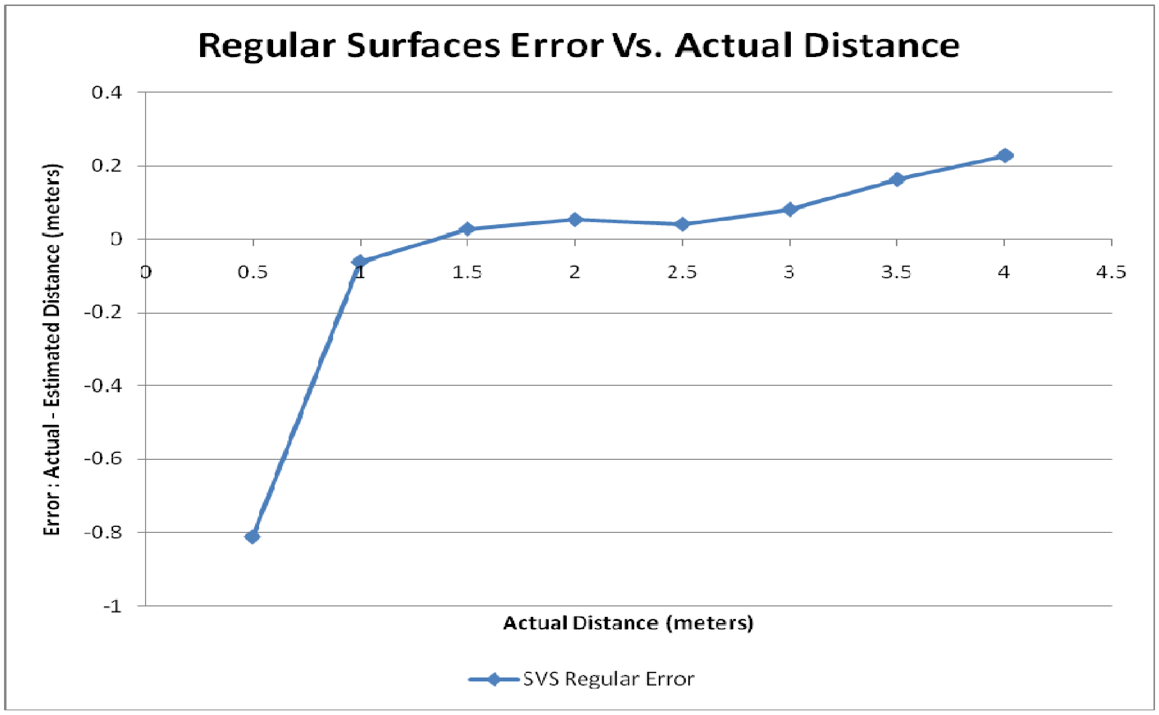


Figure 4.4 Graph for error for regular surfaces vs actual distance

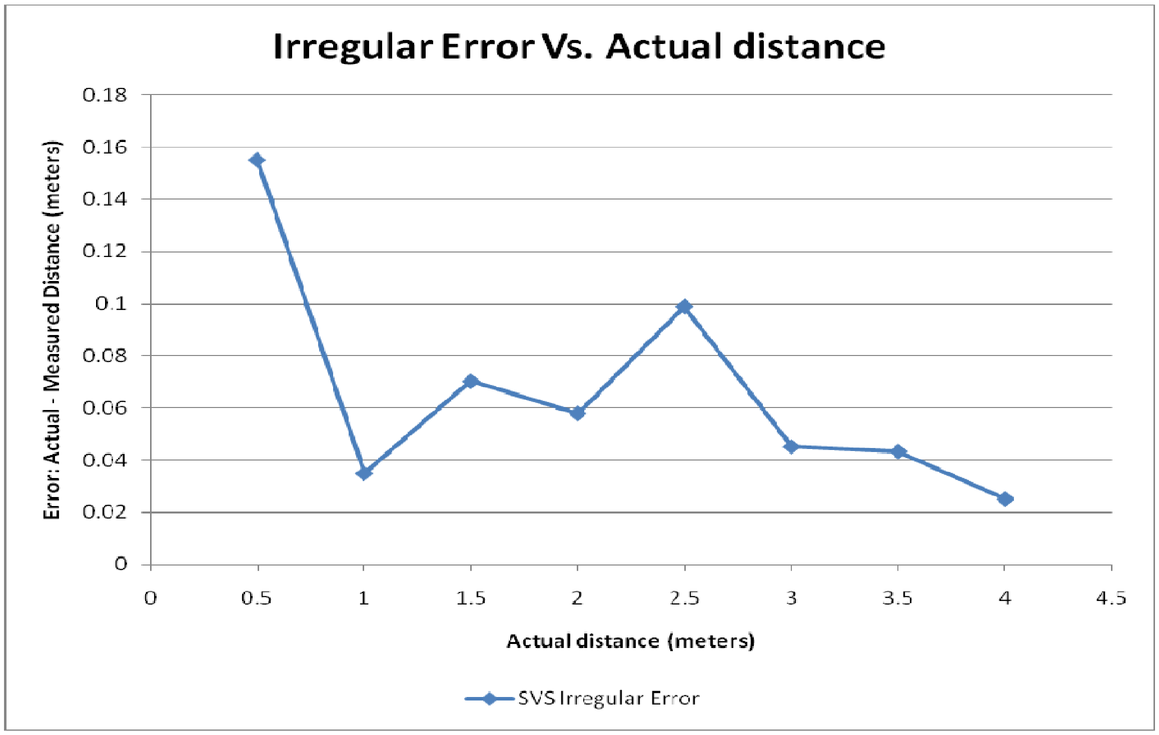


Figure 4.5 Graph for error for irregular surfaces vs actual distance

usage where hover is required above 0.5 meters to 5 meters it is quite reliable. The camera shows some problems with calibration, with each calibration taking up to 15-20 minutes. It loses its calibration settings quickly, due to the lenses losing focus under sudden movement or shock. If the camera is to be fixed on a quadrotor, it has to sturdy enough to overcome vibrations and sudden movements which may cause it to lose focus. As such, un-calibrated measurements of distances are required to check whether the errors are small enough for it to be considered as a possible sensor.

# CHAPTER FIVE

# CONCLUSIONS AND RECOMMENDATIONS

## Conclusion

This thesis describes a PID position control method which uses the actual position, velocity and orientation, found via sensors, of the quadrotor as feedback to control and reach a desired position. Simulations were conducted to verify the validity of the control approach. Attitude control simulations and experiments were conducted to confirm satisfactory results.

Position control was simulated by using quadrotor dynamics feedback to the system. The user has only to specify the position of the quadrotor and the controller works to achieve it. It does so by generating a trajectory of commanded position from the desired position and actual position, using a ramp function, which the quadrotor then follows by trying to control the velocity of the quadrotor. Trying to control the velocity of the quadrotor may not be a logical step as the dynamics show acceleration in a certain axis on orientation of the quadrotor in that axis. However, due to inability to measure accelerations of the body, velocity of the body is used. Thus essentially the quadrotor tries to maintain a certain velocity, which it does by increasing and decreasing desired angles in that axis. Due to velocity dynamics being slower than the orientation dynamics, the transfer function between the orientation and velocity can be thought as a gain unity block.

The simulation shows the desired trajectories being followed and being achieved and being maintained in all three axes. After achieving the positions the quadrotor maintains its position until another trajectory is specified. However for the actual experiments, problems with maintaining the thrust of the quadrotor, large GPS errors and varying characteristics of the motors could not allow for successful implementation.

The attitude controller part was equally important as the positions are finally controlled and attained using orientation control. The velocity control part could actually be removed and position could be attained by just using the angles. However, including velocity control provides greater stability to the system by providing feedback. The simulations conducted showed that angles could follow the desired angles.

Experiments were performed to check the hover conditions of the quadrotor, using the MIDG II sensor orientation data as feedback. More tests were performed at various frequencies with a sine wave as input to simulate slow or fast change of desired angles. The results also provide information about the response time of the system. The attitude controller allows for a user to easily control the quadrotor using a joystick or though a trajectory as long as the thrust is provided manually. The problem with the controller is the pitch reaching $\pm \frac{\pi}{2}$ radians. However under normal flight conditions, these values are not reached.

## Recommendations and Future Work

Groundwork has been laid down for future projects for testing the position controller. Due to several constraints, it is recommended to change a number of aspects related to the project. Some recommendations are critical for complete autonomous flight and other recommendations can pave the way for a smoother overall flight.

The GPS sensor on board does not provide reliable results to be used as a complete position and velocity sensor on its own. The large errors may be reduced by using a Differential Global Positioning system as a means of auto correcting and considerably reducing the errors. The errors from the GPS can cause sudden changes to the actual position estimates of the quadrotor making it suddenly increase or decrease the thrust or produce more pitch or roll. If sudden changes happen within a small span of time, it could be detrimental to the overall system stability and may cause wild oscillations from which the quadrotor may not recover.

Even with a Differential GPS (DGPS) system, the quadrotor is bound to have sufficient position and velocity errors that can cause problems in flight. Another sensor such as an ultrasonic range finder, IRs (Infrareds) or the stereo vision camera is recommended to offset any possible errors of Differential GPS. However, these sensors have their own errors and a Kalman filter would be required to give get accurate results.

The MIDG sensor sends data at a rate of 50 Hz, with GPS data coming in at 5 Hz, which is Kalman filtered to also provide position and  velocity at 50 Hz. Normally for aerial vehicles, a frequency rate of above 200 Hz is expected. Normal usage for such sensors in quadrotor involves much higher update rates. Higher update rates shifts to smoother and stable flight of the quadrotor.

One of the foremost problems with controlling the DraganFlyer is the PCB board on it. There is no information available on how the board calculates torques and thrust commands to the motors, how the inner loop control works and whether there is any saturation on the inner loop. It was attempted to contact DraganFlyer for information regarding the inner loop, but information on the control was not revealed. Also the gyros used by the control for its own stabilization are not good quality and will lead to errors in the loop itself. If it is possible to remove the PCB board altogether and build a new one, it would extensively reduce the errors in orientation and the response of the system. Instead of settling into a sine wave around zero, the angles may be actually reduced to zero under proper conditions.

Another important factor is the response time of the system, which we calculated to be about 0.5 seconds. The ideal response time of such a system is between 0.1-0.15 seconds due to the high speeds it can attain. At sudden changes in position, velocity and orientation, the quadrotor needs to respond quickly to the change in conditions so as avoid crashes and unstable control. Till then an over damped system would be beneficial.

Also at the same time thrust is also provided through the PCB board. If the required thrust to be outputted to the motors can be modeled, then position control can be possible even if GPS errors have to be taken into account. The thrust given to the quadrotor is in voltage form, while it is calculated by the controller in Newton. A correlation has to be found between the two which matches the required thrust values to the output voltage. With increase in thrust, the zero values of the roll, pitch and yaw angles also change due to the non-linearities in the motor. Initially the motors were assumed to have the same characteristics but after experiments, it was found that with increase in thrust, the individual motors outputted different voltages for the same input voltage. If a different PCB board is used, a separate controller for each motor can be implemented for better response of the system.

The motors used are carbon brushed motors [14] which are not very efficient and heat up and fail quickly. In a span of six months, we went through 4 motors, due to which new zero values for roll, pitch and yaw had to be tested as the motor characteristics changed with each individual motor. Brushless motor are much more efficient and allows less power consumption. This may possibly increase DraganFlyer flight time from the current 4 minutes to 6-8 minutes at hover.

# APPENDIX

## Sensors

### MIDG II Modes and Setup

The MIDG II has three main modes of operation depending upon user configuration and internal operating criteria. These modes are: IMU, VG and INS mode. Depending on whether user configuration is present or not, the MIDG II uses the default configuration on its non-volatile memory. The default mode of operation for the MIDG II is the INS mode.

The IMU mode is the most basic mode, providing angular rate, acceleration and magnetic field calibrated values.GPS raw values are also available albeit without any filtering done on them, with a rate of 5 Hz. The VG (Vertical Gyro) mode allows for orientations to be estimated using integration of rate sensors along with Kalman filtering of magnetometer and accelerometer data. In INS mode, position, velocity and orientation estimated values are available at 50 Hz, with error corrected angular rates and accelerations.

The MIDG is connected to the computer either wirelessly or through a wired connection. Wirelessly, it uses the XTend RF Modem to transfer the data back to the computer through another XTend Modem, with signals being received in RS -232 forms. If connected with a wired connection, the cable for connection uses a serial chip converter to convert the RS-422 data to RS -232 signal that can be used by the computer.

The data received from the MIDG II by the data packets are converted into a single data structure using a client/server program using software provided by Microbotics. In Windows, the data can be saved from the serial port onto a file which can be parsed using a Microbotics Program [35]. In QNX, a server program, MIDGServer, runs and receives data, parsing them and storing them into a data structure. A client program reads off the shared memory and provides the data to the controller.

## Drift in Angles

MIDG II is a Micro-Electro-Mechanical Sensor (MEMS) , whose gyros contain a vibrating mass that generate a force when rotated due to Coriolis Forces. By measuring these forces, angular rates can be determined. These gyros can measure angular rates in roll, pitch and yaw directions. To find the orientation, these angular rates have to be integrated using an angular rate bias that slowly varies over time. However there are still errors in the readings which increase with time due to factor including sensor bias, noise and integration errors. For accurate readings, these errors have to be reduced as much as possible.

Roll and pitch angles can be corrected using accelerometers are secondary method of measurements. Accelerometers are used to check whether the MIDG II is level, as its readings point one gravity in the downward z direction according to the earth's inertia frame. The difference in the sensor angle and the known inertia angle, gives us the angular bias for roll and pitch allowing for their drift correction.

The yaw angle, being independent of the gravity vector, requires another sensor for its correction. The 3 axis magnetometers are used for correcting the yaw drift. The magnetometers measure the earth's magnetic field in the x, y and z direction. By projecting the vector of magnitude in the x-y plane, the yaw angle can be found relative to the North Pole. Using the two known orientation angles in conjunction with a known magnitude, the third orientation angle can be found. However when the magnetic field and gravitational lined line up, meaning there is a pitch of $\pm\frac{\pi}{2}$, there is a singularity and one orientation cannot be measured.

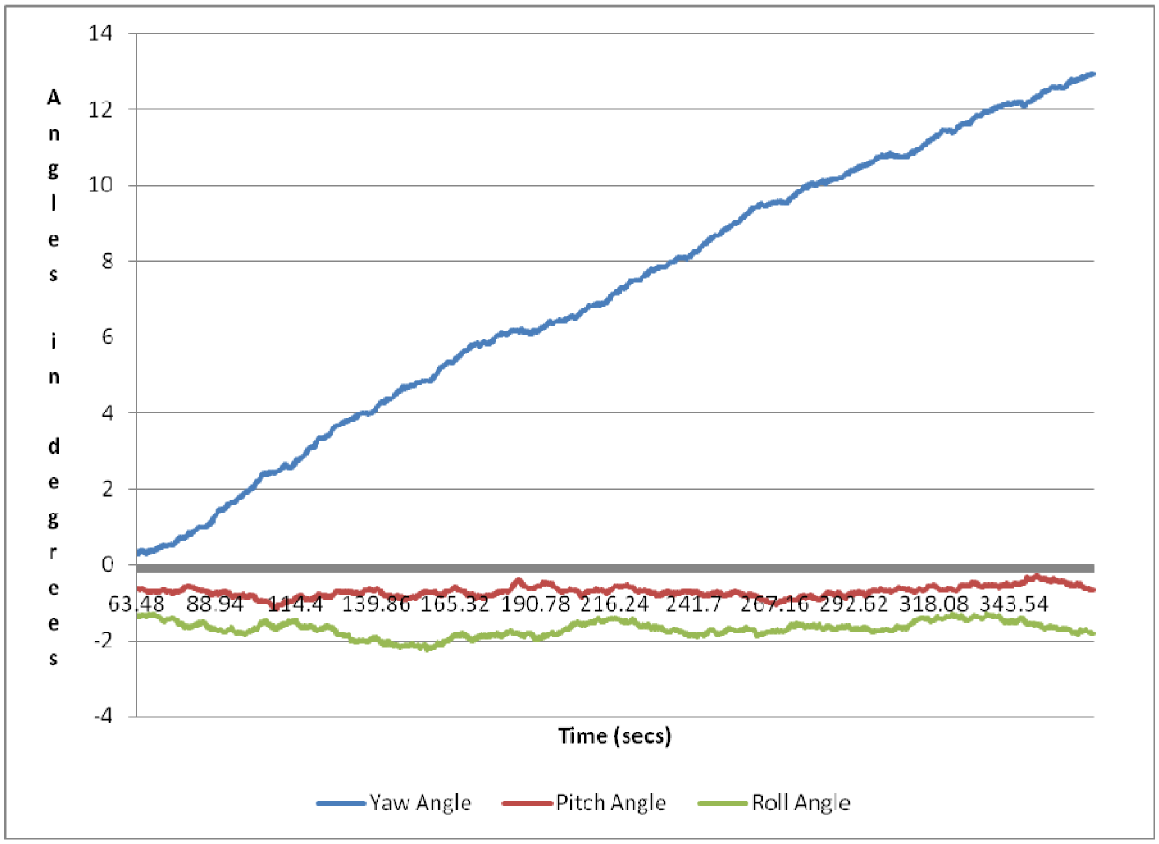


Figure A-1 Drift angle measurements

Here the assumption is made the earth's magnetic field is the only magnetic field acting in the region. With any shift in magnetic field, the yaw heading changes in only a few seconds. Figure A-1 illustrates the fact, where the roll and pitch angles are like sine waves which correct themselves after a few seconds but yaw continues to drift albeit slowly.

## Global Positioning System

The GPS results were measured to check the validity of the results received from the sensor. The quadrotor was kept stationary on the ground for some time and the position and velocity results were noted. The power to the MIDG and the GPS was turned off and switched back again. A square of 45 meters side length was then followed going clockwise, first going towards north, while keeping the quadrotor with the GPS antenna at a constant height. All values of the GPS shown here are in are in ENU (East North Up) format as the MIDG II uses ENU as the default configuration.

Thus the x axis values represent the east position, y axis values represent the north position and z axis represents the up position with respective to its original starting point. Figure A-2 shows the filtered positions. For the stationary quadrotor tests, the results are mostly within an error range of 1.5 meters for the filtered Up position with an offset of approximately 12.5 meters which can be zeroed out. North and East positions have an error range of approximately 1 meter and 0.5 meters respectively. This is the best result displayed received through the GPS sensor in all the tests.

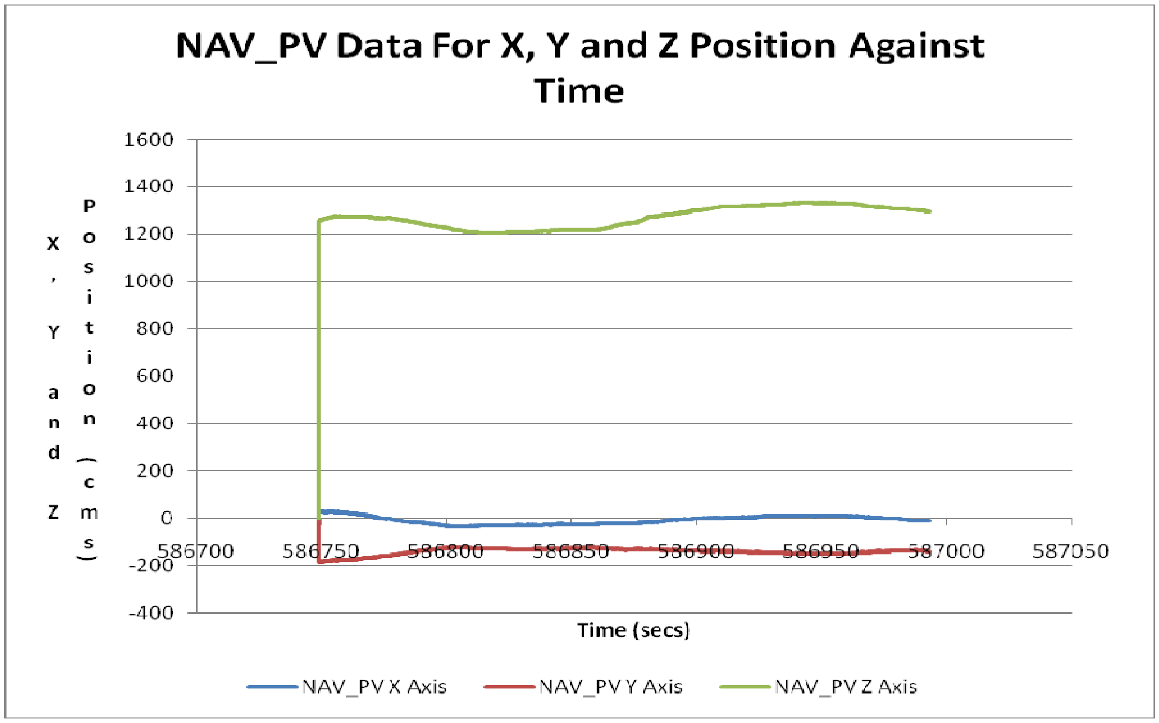


Figure A-2 Stationary quadrotor position test

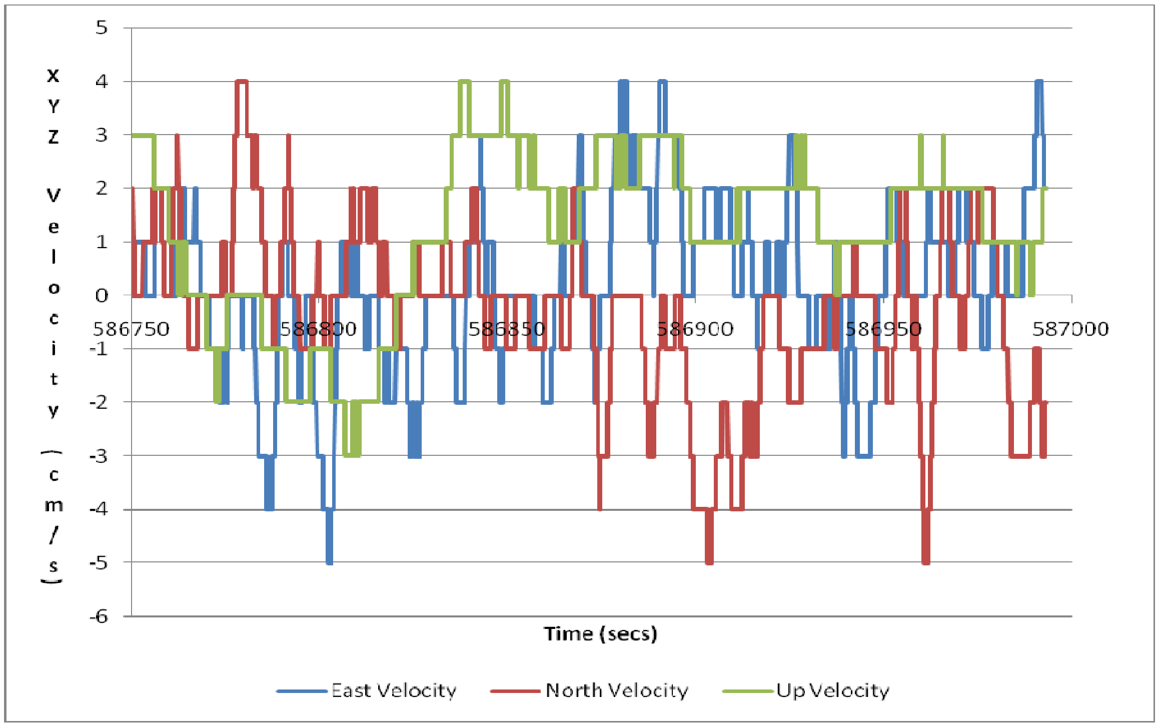


Figure A-3 Stationary quadrotor velocity test

Figure A-3 shows the stationary quadrotor velocity test. The filtered GPS velocity show small errors for a stationary quadrotor and show a maximum error of 5 cms/sec for each of the three directions.

The results for the moving quadrotor are given in figure A-4. The path which was followed was rigorously checked for distance and accuracy to north and east directions.

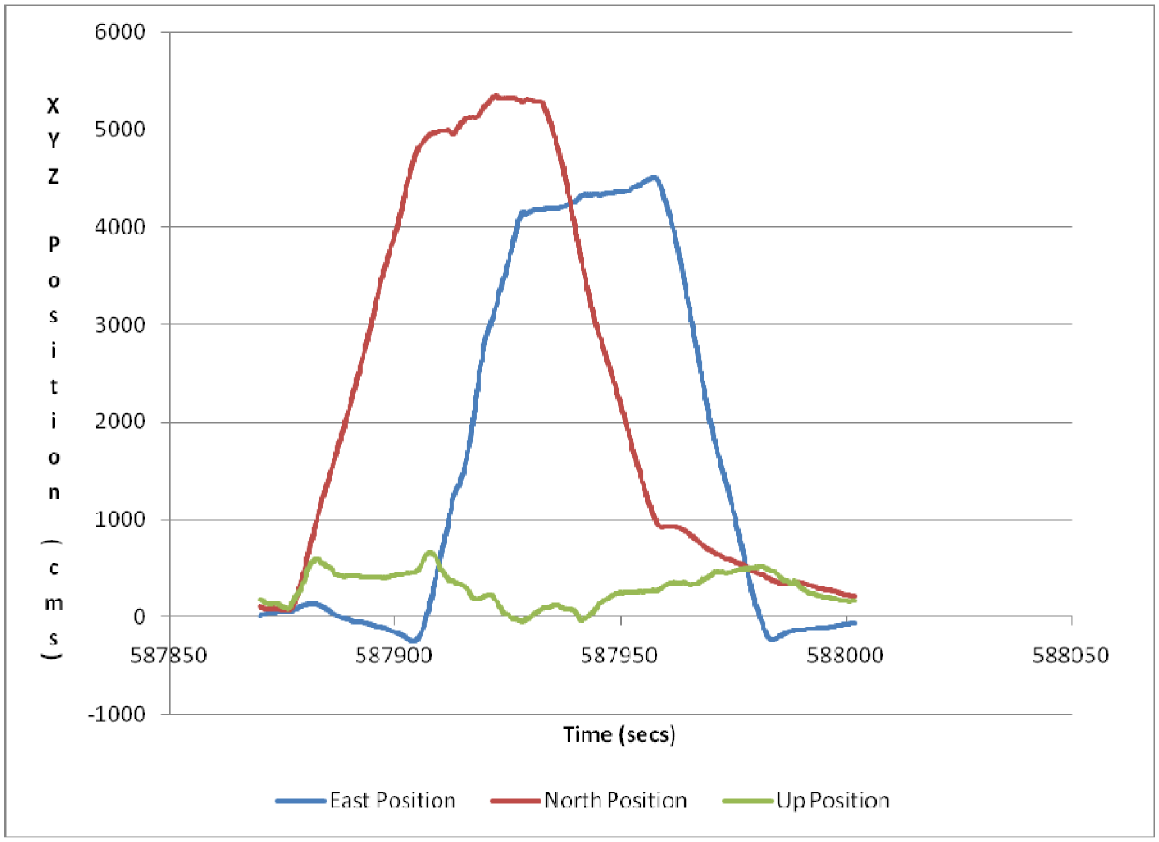


Figure A-4 Moving quadrotor position test

The North axis results are off by some margin to what was followed on the ground and show an error of 5-10 meters on reaching the required 45 meters. For East position, the margin of error was lesser and it stayed within a margin of 4 meters. However, the ost adversely affected was the z position with errors ranging from 0 to 12 meters. Tests like these were performed some more times with similar results, with East and North position error results varying between 4-10 meters and z position error results varying between 6-15 meters with time. This result was performed within a span of 150 seconds.

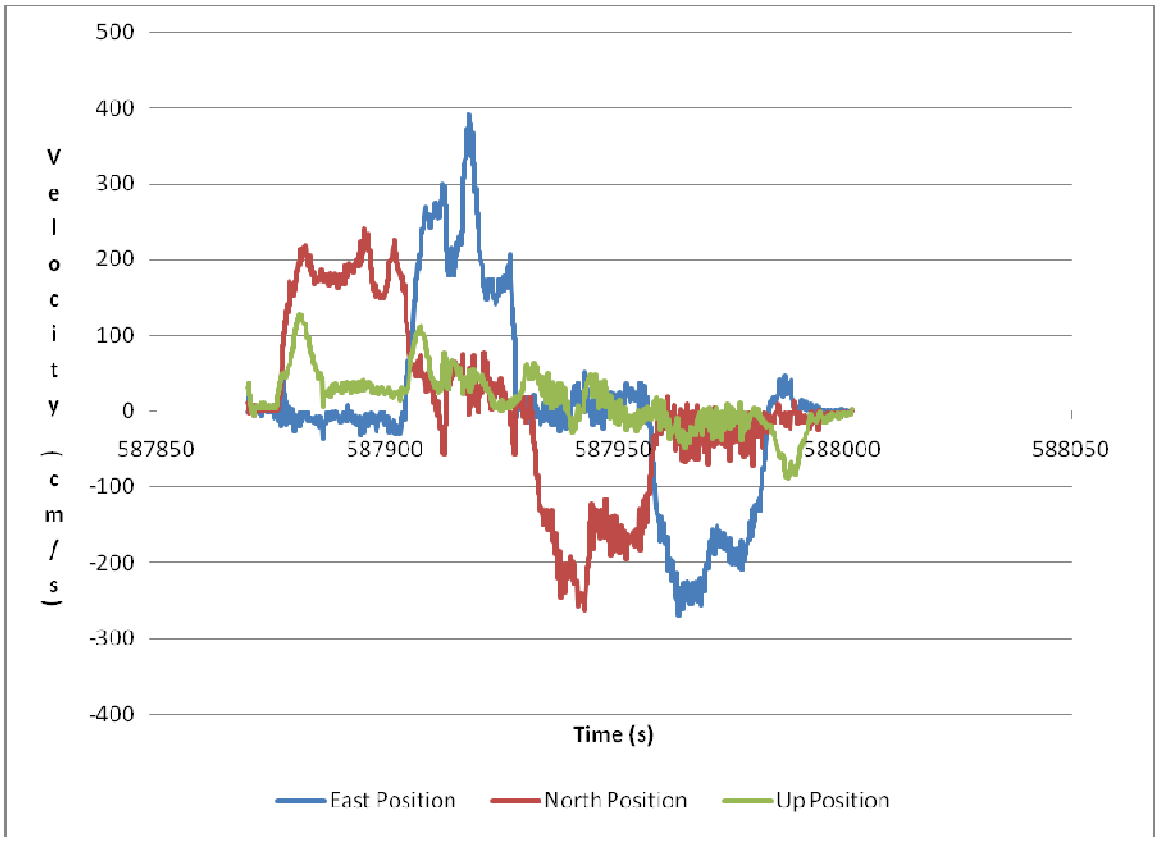


Figure A-5 Moving quadrotor velocity test

The velocity results are much better than the position results and just by looking at the graph, it can be observed that first the quadrotor moved north, moved east, moved south and then moved west. The velocity estimations of the quadrotor seem to be a better representation of motion than positions. If possible, it would be beneficial to use the GPS velocity estimates more reliably than the position estimates and a differential GPS signal should be used for the quadrotor to improve the position estimates.

# REFERENCES

[1] A. Gessow, G. Myers, Aerodynamics of the helicopter, Frederick Ungar Publishing Co, New York, third edition, 1967.

[2] G. Hoffmann, D. Rajnarayan, S. Waslander, D. Dostal, J. Jang, and C.Tomlin "The Stanford Testbed of Autonomous Rotorcraft for MultiAgent Control (STARMAC)", Proceedings of the 23rd Digital AvionicsSystems Conference, Salt Lake City, Utah, pp. 12.E.4-12.E.10, November, 2004.

[3] Gabriel Hoffmann, Haomiao Huang, Steven Waslander, Claire Tomlin, AIAA Paper Number: AIAA-2007-6461, AIAA Guidance, Navigation and Control Conference and Exhibit, Hilton Head, South Carolina, Aug. 20-23, 2007.

[4] J. Jang and C. Tomlin, "Longitudinal Stability Augmentation System Design for the DragonFly UAV using a Single GPS Receiver", Proceedings of the AIAA Guidance, Navigation, and Control Conference, Austin, Texas, AIAA Paper Number 2003-5592, August 2003.

[5] Jonathan P. How, Brett Bethke, Adrian Frank, Daniel Dale, and John Vian, "Real-Time Indoor Autonomous Vehicle Test Environment", Control Systems Magazine, IEEE Publication Date: April 2008, Volume: 28, Issue: 2

[6] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski, "Dynamic Modelling and Configuration Stabilization for an X-4 Flyer," Proceedings of the IFAC World Congress, Barcelona, Spain, July 2002.

[7] Zak Sarris, "Survey Of UAV Applications In Civil Markets" http://med.ee.nd.edu/MED9/Papers/Aerial_vehicles/med01-164.pdf

[8] Andy Neff, " Linear and Non-Linear Control of a Quadrotor UAV", http://etd.lib.clemson.edu/documents/1181251774/umi-clemson-1183.pdf

[9] DraganFlyer X-Pro, http://www.rctoys.com/draganflyerxpro.php

[10] M. W. Spong and M. Vidyasagar, Robot Dynamics and Control, John Wiley and Sons, Inc: New York, NY, 1989.

[11] Sørensen, M ; Kjaergaard, M ; Bjørn, J, "Autonomous hover flight for a quad rotor helicopter: Linear Quadratic Controller, Piecewise Affine -Hybrid systems controller", http://projekter.aau.dk/projekter/research/ autonomous_hover_flight_for_ a_quad_rotor_helicopter(9932148)/

[12] Jesper Haukrogh, Ole Binderup, Sigurgeir Gislason, "Platform development,Sensor modelling and Estimation of a Draganflyer X-Pro", http://www.control.aau.dk/uav/reports/07gr1037b/07gr1037b_student_report.pdf

[13] Neil G. Johnson, "Vision-Assisted Control Of A Hovering Air Vehicle In An Indoor Setting", http://contentdm.lib.byu.edu/ETD/image/etd2430.pdf

[14] RS-545SH Carbon Brush Motors, http://www.mabuchi-motor.co.jp/cgi-bin/catalog/e_catalog.cgi?CAT_ID=rs_545sh

[15] IRL 14040 40V Single N-channel HexFET Power MOSFET in a TO-220AB Package, http://www.digchip.com/datasheets/parts/datasheet/232/IRL1404.php

[16] Park, S., Won, D.H., Kang, M.S., Kim, T.J., Lee, H.G., Kwon, S.J., Div. of Appl. Robot Technol., Korea Inst. of Ind. Technol., South Korea; "RIC (robust internal-loop compensator) based flight control of a quad-rotor type UAV", Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference, Publication Date: 2-6 Aug. 2005, On page(s): 3542- 3547

[17] Castillo, P., Dzul, A., Lozano, R.; Centre de Recherche de R.lieu, Univ. de Technol. de Compiegne, France; "Real-time stabilization and tracking of a four-rotor mini rotorcraft", Control Systems Technology, IEEE Transactions, Publication Date: July 2004, Volume: 12, Issue: 4, On page(s): 510- 516

[18] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor", Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference, Oct. 29 2007-Nov. 2 2007, pp. 153-158. 19

[19] Videre Systems, http://www.videredesign.com/

[20] Small vision system, http://www.videredesign.com/vision/svs.htm

[21] V. Chitrakaran, D. Dawson, J. Chen, and M. Feemster, "Vision Assisted Autonomous Landing of an Unmanned Aerial Vehicle", Proceedings of the IEEE Conf. on Decision and Control, Seville, Spain, pp. 1465 - 1470, December, 2005.

[22] JR PROPO PCM9XII R/C controller, MacGregor Industries http://www.macgregor.co.uk/radio/PCM9X2.htm

[23] Nicolas Guenard, Tarek Hamel, Robert Mahony, "A practical Visual Servo Control for a Unmanned Aerial Vehicle", 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10-14 April 2007

[24] T. I. Fossen, Marine Control Systems : Guidance, Navigation, and Control of Ships, Rigs, and Underwater Vehicles, Marine Cybernetics, 2002.

[25] Steven, Lewis, Aircraft simulation and Control, J. Wiley & Sons, 2003

[26] QNX Software Systems, http://www.qnx.com/

[27] QMotor Real-Time Control Environment, http://www.ece.clemson.edu /crb/research/realtimesoftware/qmotor/index.htm

[28] MIDG II INS/GPS Sensor, http://www.microboticsinc.com/ins_gps.html

[29] XTend RS-232/RS-485 RF Modem, http://www.maxstream.net/products/xtend/rf-modem-rs232.php

[30] Microbotics Message Specification, http://www.microboticsinc.com/midg _files/MIDG%20II%20Message%20Specification%20FW2_2.pdf

[31] SRI International, Small Vision System, http://www.ai.sri.com/software/SVS

[32] Stereo On Chip Calibration, http://www.videredesign.com/docs/stoc-1.3.pdf , pages 25-26.

[33] SVS Calibration, http://www.videredesign.com/vision/svs_calibration.htm

[34] MIDG Modes of Operation, http://www.microboticsinc.com/midg_files /AN001OpModes.pdf

[35] Microbotics Data Parser, http://www.microboticsinc.com/downloads/MIDG @/M2parsers.zip