Faculty of Aerospace Engineering

Dynamic Modelling and State Estimation of a High Speed Racing Drone

Nishant Nipa Patel

November 5, 2020

Delft University of Technology

Challenge the future

Dynamic Modelling and State Estimation of a High Speed Racing Drone

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

Nishant Nipa Patel

November 5, 2020

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Copyright © Nishant Nipa Patel
All rights reserved.

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Dynamic Modelling and State Estimation of a High Speed Racing Drone"** by **Nishant Nipa Patel** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: November 5, 2020

Readers:

Prof. Dr. G. C. H. E. de Croon

Dr. J. Guo

Ir. C. de Wagter

Acknowledgements

I would like to dedicate this project to everyone who has been an important part of my life, especially during the last few years, which have shaped me to be the person I am today.

Firstly, I would like to thank my mother, whose love, support, and encouragement enabled me to pursue my master's in a place so far from home.

I would like to thank my supervisors Guido de Croon and Yingfu Xu for their guidance throughout the course of my thesis. Christophe and Freek helped me in obtaining the datasets I needed for this research, but also left me in awe with their flying skills that I hope to replicate some day. I would also like to thank Erik, Nilay, and Rohan for their constant help and some great conversations.

Despite beginning my master's alone in a foreign land, I ended up with great friendships within Delft and beyond. Yagiz, Hrishikesh and Arjav were amazing companions throughout these two years. Project Talaria didn't just leave me with great engineering skills but also long lasting memories and some awesome mates in Yash, Roelof, Patrick and many more. I must mention Romy, Soham, Darshit, and of course my best friends from Jadda, who have been there for me thick and thin. I would also like to thank my girlfriend Nina for keeping me motivated to complete this project during such a pandemic.

Nishant Patel
Delft, November 2020

Abstract

Autonomous drone racing has taken a turn for the better in recent years. Drones are becoming faster and implementing better state-of-the-art control techniques to overcome different challenges. With advancements in the fields of computer vision, machine learning, and artificial intelligence, the final goal of autonomous drones is to be quicker than human-piloted racing drones. Increasing the speed of autonomous drones increases the risks associated with flying them. Time-optimal control algorithms have been identified as a method of implementing aggressive maneuvers to fly drones at high speeds throughout the course of the race. These methods require precise state-estimates. This research work identifies a model for the controller. The work also includes an implementation of a state estimation model with drag compensation, also merging a pre-existing refined thrust model with Coriolis effects. With the idea of developing a state estimation model for a racing drone, the model is improved to include flight envelopes involving motor saturations.

Acronyms

COM	Center of Mass
DOF	Degrees of Freedom
EKF	Extended Kalman Filter
ESC	Electronic Speed Controller
FPV	First Person View
GPS	Global Positioning System
IMU	Inertial Measurement Unit
MAV	Micro Aerial Vehicle
MAVLab	Micro Aerial Vehicle Laboratory
MoCap	Motion Capture
PID	Proportional Integral Derivative
RMSE	Root Mean Square Error
RPM	Revolutions Per Minute
SLAM	Simultaneous Localisation and Mapping
UAV	Unmanned Aerial Vehicle
VIO	Visual Inertial Odometry

List of Symbols

E_b	body frame of reference
E_i	inertial frame of reference
$\boldsymbol{\Omega}$, p, q, r	angular velocities of the vehicle in the body frame
$\boldsymbol{\xi}$, x, y, z	position of the vehicle in the cartesian coordinate system
$\boldsymbol{q_{wb}}$, q_w, q_x, q_y, q_z	quaternions representing the orientation of the quadrotor
ω_i	rotation speed of the i^{th} rotor
$\mathbf{I}$, I_{xx}, I_{yy}, I_{zz}	moment of inertias of the vehicle
$\mathbf{J}$	transformation matrix for angular velocities in terms of the euler angles
$\mathbf{R}$	rotation matrix in terms of the euler angles
$\mathbf{V}$, u, v, w	linear velocities of the vehicle in the body frame
g	acceleration due to gravity
m	total mass of the vehicle
$\boldsymbol{\eta}$, ϕ, θ, ψ	euler angles representing the attitude of the quadrotor

Contents

List of Figures

Chapter 1

Introduction

In the present world, UAVs are finding applications in the fields of recreation (aerial photography), commercial (delivery drones), industrial (inspection drones) and even the military (search and rescue operations). For better portability and lower costs, the demand for smaller and lighter UAVs has led to increased research in MAVs. Owing to their small size, the quality of the sensors being used, and limitations in the processing capabilities, autonomous flights are a challenge for MAVs.

In the recreational field, an e-sport known as FPV drone racing is becoming increasingly popular. The objective of a drone race is similar to any other race - to complete a pre-set course in the fastest time possible. The scientific advancements in research areas like artificial intelligence, computer vision, control systems and machine learning have raised the challenge of designing an autonomous drone that can complete a race course faster than a human pilot.

Several challenges are involved in autonomous drone racing (Moon et al. [2019]). The drone needs to perform short and agile flight maneuvers using only on-board sensors. The drones are required to pass through several gates and therefore, require good state estimates and control strategies. While GPS combined with on-board inertial sensors is a useful way of estimating the pose of the drone and detecting the gates whose positions are unknown, it is not an option for indoor drone racing. Therefore, a front-facing camera is used as the primary on-board sensor along with IMUs to detect the gates as well as provide a pose estimate for the MAV along with data obtained. Some races might also include moving gates and other obstacles for which the implemented control strategy should be more robust and the sensors and state estimates, more accurate.

A major difference between a human pilot and current autonomous control strategies is the capability to perform aggressive maneuvers. Aggressive maneuvers are important to take quick actions at high speeds. Time-optimal control problems have been identified as a solution to performing aggressive maneuvers similar to human pilots. This requires estimates of the states of the drone which can be calculated by modelling the dynamics of the quadrotor. While there are a multitude of theoretical dynamic models for quadrotors (Das et al. [2009], Pounds et al. [2010]), assumptions are made to simplify the system, due to which the state estimates are quite inaccurate when compared to the measured states. Assumptions like the exclusion of drag forces and near-hover conditions cannot be used in the case of drone racing. An alternative method of modelling considered is system identification. System identification principles can be used to create a dynamic model directly from the flight data by measuring the control inputs and the resulting kinematic response of the MAV (Gremillion and Humbert [2010]). A motion tracking system can be used to obtain the position, velocity and attitudes of the drone in real-time. The data can be studied and used to correct the state estimates and design a more accurate state estimation model.

The only way to change the thrust or the attitude of the drone is to alter the speed of the motors. The motors have their physical limits and thus get saturated at a certain point. The models mentioned above is generally for low speed flights. Therefore, the thesis modifies the estimation model obtained to include saturation-based flights of the quadrotor

1-1 Motivation and Research Objectives

The MAVLab conducts extensive research in the field of autonomous control for MAVs. This ensures that any research within this lab allows one to work with a combination of hardware and software, polishing the basic skills in both these areas. The team of students, faculty and researchers also participates in various drone racing leagues around the world, completing a podium finish in a few. In order to win a race, the trajectory generated by the quadrotor must be time-optimal. This requires accurate state estimates to ensure the drone does not crash into a gate in its path. Improving the estimation by combining real-time in-flight measurements with theoretical principles could provide the required accuracy.

Time-optimal trajectories might also require a drone to saturate its motors. Using the motors at saturation might lead to unwanted changes in altitude and unnecessary drifts from the path. If the response of the quadrotor can be calculated for motor saturations in the form of thrusts and rotation rates, it could be used to estimate its velocity.

From this motivation, the research goal is framed as the following question.

Can system identification principles be used to obtain a model using real-time flight data to estimate the states of the drone with a higher accuracy than existing methods?

This question can be further divided into a set of sub-questions to be answered in this study.

- *What kind of estimation model is best suited for high-speed flights?*
- *How accurate is the predicted data when compared with the groundtruth?*
- *Does the dependency of certain parameters change as the speed of the quadrotor increases?*
- *Are the estimations obtained from this model better than previously implemented state estimation methods and why?*

The idea of this study is to implement a state estimation method based on real-time flight data. The estimation model must account for the dynamics of the drone as well as the implemented rate controller. Finally, the state estimates of the model will be compared for a quadrotor flying at saturations.

1-2 Structure of this work

The scientific paper, consisting of the main contributions of the thesis, is presented in Part I. It can be read as an independent paper with its own introduction, description of quadrotor dynamics and aerodynamic effects, state and parameter estimation models, and the results obtained including a comparative study with pre-existing state estimation models.

Part II includes a detailed study of the relevant literature. Chapter 2 describes quadrotor dynamics and the aerodynamic effects experienced by a quadrotor in flight. An overview of system identification is presented in chapter 3. The basic idea of this research along with examples of research work relevant to this thesis are described in chapter 4. Chapter 5 goes into detail about a specific method of system identification known as parameter estimation. A summary of the literature study is provided in chapter 6.

Part III consists of the appendix to support various aspects of the study. Appendix A provides a detailed description of the general approach in a system identification study.

Part I

Scientific Paper

Position Estimation of a Racing Drone with compensated Aerodynamic Effects

Nishant N. Patel[*], Yingfu Xu[†], Guido C.H.E. de Croon[†]
Department of Control and Simulation, Faculty of Aerospace Engineering
Delft University of Technology, The Netherlands

***Abstract*—Drone racing is a popular e-sport where drones fly through a pre-set course at high speeds. With rapid improvements in technology, autonomous drone racing is beginning to challenge experienced human pilots. Time-optimal control algorithms have been identified as a method of enabling the drone to perform aggressive maneuvers to increase its speed throughout the race. This requires good state estimates for the drone to ensure that it does not crash into a gate at high speeds. This paper identifies models for the rate controller and a state estimator. The estimator uses a pre-existing "refined thrust model" to estimate the accelerations. The results depict that using a refined thrust model could reduce the errors in position estimation as compared to a "standard thrust model" or using raw accelerometer measurements. The position estimations are further improved by compensating for drag and Coriolis effects.**

I. Introduction

In the present world, UAVs are finding applications in recreation (aerial photography), commerce (delivery drones), industry (inspection drones) and even the military (search and rescue operations). For better portability and lower costs, the demand for smaller and lighter UAVs has led to increased research in MAVs. Owing to their small size, the quality of the sensors being used, and limitations in their processing capabilities, autonomous flights are a challenge for MAVs.

In the recreational field, an e-sport known as FPV drone racing is rapidly gaining popularity. The objective of a drone race is similar to any other race - to complete a pre-set course in the shortest possible time. The scientific advancements in research areas like artificial intelligence, computer vision, control systems, and machine learning have raised the challenge of autonomous drone racing. While research has enabled drones to complete race courses, the new challenge is competing with experienced human pilots.

Several challenges are involved in autonomous drone racing [1]. The drone needs to perform short and agile flight maneuvers using only on-board sensors. They must pass through several gates and therefore, require good state estimates and control strategies. While GPS combined with on-board inertial sensors is a useful way of estimating the pose of the drone and detecting the gates whose positions are unknown, it is not an option for indoor drone racing. Therefore, a front-facing camera is used as the primary on-board sensor along with IMUs to detect the gates as well as provide a pose estimate for the MAV along with data obtained. Some races might also include moving gates and other obstacles for which the implemented control strategy should be more robust and the sensors and state estimates, more accurate.

A major difference between a human pilot and current autonomous control strategies is the capability to perform aggressive maneuvers. Aggressive maneuvers are the result of quick actions like sudden large accelerations or high rotational rates. Time-optimal control problems have been identified as a method of performing aggressive maneuvers. Optimal control techniques rely on the estimates of the states of the drone, computed by modelling the dynamics of the quadrotor.

While there are a multitude of theoretical dynamic models for quadrotors [2], [3], assumptions are made to simplify the system. These assumptions use data from flight regimes where aerodynamic effects like drag, blade flapping, thrust variation are minimal and can result in linear models. Since race drones fly at high speeds, ignoring these aerodynamic effects lead to inaccurate estimates when compared to the measured states. System identification principles can be used to understand the dynamics better and create a dynamic model using the measured control inputs and the kinematics of the MAV ([4]). Since the flight tests are performed indoors, a motion tracking system is used to obtain the position, velocity and attitudes of the drone in real-time. The data from the on-board sensors and the motion capture system can be studied and used to obtain a more accurate state estimation model.

The estimation model implemented in this paper is divided into two subsystems. The first subsystem maps the inputs and states of the drone (measured by on-board sensors) to the rotation speeds of the motors. The sub-system identified uses the inputs and measured states to calculate the possibility of motor saturations and corrects the estimates to provide values within the physical limits of the motors. The second subsystem uses the motor speeds and current state of the drone to estimate the position of the quadrotor. This technique uses a refined thrust model as well as compensates for Coriolis effects to reduce the errors in estimation. The results from the proposed model are compared to different estimation methods in order to depict how each additional change makes an improvement.

The outline for the rest of the paper is as follows. Section II describes the dynamics of the quadrotor as well as the different aerodynamic effects on it. Section III gives a brief overview of the quadrotor used in the experiments, and describes the

[*]MSc. student, [†]Supervisors

data acquired through the set of flight tests. In section IV, the unknown parameters in the model are computed. The results of the estimation model are validated on several datasets in section V.

II. Quadrotor Modelling

The coordinates of the system are described with an orthonormal basis in either the world frame E_w or in the body frame E_b. The position vector is described by the vector $\boldsymbol{\xi} = \begin{bmatrix} x & y & z \end{bmatrix}^T \in E_i$. The orientation vector is described as $\boldsymbol{\eta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \in E_i$. The linear and angular velocity vectors are defined in the body frame as $\boldsymbol{V} = \begin{bmatrix} u & v & w \end{bmatrix}^T \in E_b$ and $\boldsymbol{\Omega} = \begin{bmatrix} p & q & r \end{bmatrix}^T \in E_b$. The relations between these parameters is depicted in Equation 1, where **R** and **J** are the rotational and angular velocity transformation matrices respectively.

$$\begin{aligned} \dot{\boldsymbol{\xi}} &= \mathbf{RV} \\ \dot{\boldsymbol{\eta}} &= \mathbf{J\Omega} \end{aligned} \tag{1}$$

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & S_\phi C_\theta \\ 0 & -S_\phi & C_\phi C_\theta \end{bmatrix} \tag{2}$$

$$\mathbf{R} = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & C_\phi S_\theta & C_\phi C_\theta \end{bmatrix} \tag{3}$$

where C_i and S_i represent cos(i) and sin(i) respectively. The rotation matrix R is orthonormal and therefore, its transpose transforms linear velocities or accelerations in the body frame to the world frame.

The equations of translational motion for the quadrotor are [5]

$$\dot{\mathbf{V}} = \frac{\mathbf{T}}{m} - \mathbf{R}^T\mathbf{g} - \mathbf{\Omega} \times \mathbf{V} - \mathbf{D} \tag{4}$$

where **T** is the thrust, g is the acceleration due to gravity, and D is the drag force. The third term in the equation represents the Coriolis accelerations.

A. Concept of induced velocity

The induced velocity (also known as downwash) is the airflow perpendicular to the rotor caused by the deflecting of air by a blade. It alters the aerodynamics around the blade, thereby causing an effect in the lift as well as the drag of the system [8]. Figure 1 depicts a blade element where α denotes the angle of attack, α_∞ the effective incidence, w is the induced velocity or downwash, V is the free stream velocity, and q is their resultant vector.

When the induced flow w increases, ϵ increases, thereby reducing the effective incidence angle and in turn, the lift. Similarly, a reduction in the induced flow would increase the effective incidence and subsequently, the lift. By definition, the lift L is perpendicular to the oncoming air flow. But due to the effect of downwash, the oncoming air flow is at an angle, thereby altering the lift vector to Γ. The horizontal component of the resultant "Lift", represented as D, is termed as the induced drag.

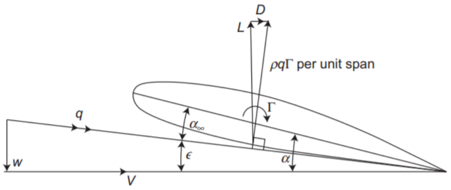


Fig. 1: Aerodynamics of a blade element [8]

B. Thrust model

Using momentum theory [6], the relationship between the thrust generated by a rotor and the square of the angular speed of the rotor ω_i is derived as

$$T_i = C_t \rho A_{r_i} r_i^2 \omega_i^2 = K_t \omega_i^2 \tag{5}$$

where for the i^{th} rotor, r_i is the radius and A$_r$ is the rotor disk area, ρ is the air density, and C_t is the thrust coefficient influenced by the rotor geometry and profile. From hereon, this will be referred to as the "standard" thrust model [7].

A more accurate thrust model is derived using blade element moment theory [9].

$$T_i = c_1 \omega_i^2 \left(c_2 \left(1 + \frac{3}{2}\mu_i^2 \right) - \lambda_i \right) \tag{6}$$

where the advanced ratio μ_i is related to the horizontal component of velocity V_{h_i} and induced velocity v_{h_i}, and the inflow ratio λ_i is related to the vertical component V_{z_i} of velocity and induced velocity v_{z_i}, as depicted in the equations below.

$$\mu_i = \frac{V_{h_i} + v_{h_i}}{r\omega_i} \tag{7}$$

$$\lambda_i = \frac{V_{z_i} + v_{z_i}}{r\omega_i} \tag{8}$$

With the assumption that the propeller is sufficiently rigid (the flapping angle is very small), $v_{h_i} \approx 0$. By rewriting v_{z_i} as a function of ω_i and V_{z_i}, equation 6 can be simplified to

$$T_i = k_t \omega_i^2 - k_z V_{z_i} \omega_i + k_h V_{h_i}^2 \tag{9}$$

where the constants k_t, k_z, and k_h are computed from experimental data.

C. Drag model

The drag force acting on the system is in different forms such as the induced drag, parasitic drag, profile drag, and translational drag. With the assumption that all flight experiments are performed indoors in an area without the influence of external wind, the different types of drag forces, excluding parasitic drag, can be lumped into a single linear function

(a) Eachine X220s FPV racing drone (b) Diatone GT-M540 FPV racing drone

Fig. 2: The quadrotors used in this research

proportional to the linear velocity of the quadrotor in the body x-y plane [10]. Parasitic drag is considered to be negligible in the flight regime used in this research.

$$\mathbf{D} = k_d \omega_s P \mathbf{R}^T \dot{\boldsymbol{\zeta}} \tag{10}$$

where k_d is the drag coefficient, ω_s is the mean of the speed of the four rotors, and P is a projection matrix

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{11}$$

III. Experimental Setup

Two quadrotors, depicted in Figure 2, are used in this research. The first quadrotor used in this experiment was the Eachine Wizard X220s. An Omnibus F4SD controller board was used for its faster CPU, and the availability of an SD card slot used for logging data from the on-board sensors. The second quadrotor used was the Diatone GT-M540 FPV racing drone, equipped with a Mamba F405 flight controller. Both the flight controllers were uploaded with the Betaflight flight controller firmware.

The reason for using two different quadrotors was that the ESCs of the Eachine would not allow the drone to saturate the motors using a 4S battery. Therefore, in order to obtain data for motor saturations, a different drone and micro-controller combination was required.

Betaflight is a rate controller, and therefore operates as the innermost loop of an autonomous flight controller. It accounts for the attitude rate measurements as well as the user inputs for rate and throttle, and returns a command (analog or digital signal) for the motor ESC (electronic speed controller).

A. *Data acquisition*

Betaflight records the values for the attitude rates and linear accelerations from the on-board IMUs. It also logs the user inputs, motor command values sent to the ESCs and in certain cases, the rpm of each rotor. The flight tests are performed indoors at the Cyberzoo in the faculty of Aerospace engineering at TU Delft, allowing the use of the Optitrack motion capture system to obtain the position coordinates and orientation of the quadrotor.

Two subsystems are drawn from the experimental data obtained. The inputs for the first subsystem are the measured states and the control inputs, and the output is the speed of each rotor. The second subsystem uses the rotor speeds as inputs to the dynamics of the quadrotor to provide estimates for the position. To validate the model, the resulting state estimates are compared to the groundtruth from Optitrack, as well as results from different estimation models.

B. *Sensor fusion*

Fusing the on-board measurements from Betaflight and the Optitrack data is essential to obtain more accurate and reliable parameters to model the quadrotor dynamics. Optitrack position measurements are highly accurate (less than 1mm errors) and do not degrade with time.

An obstacle in sensor fusion is the different sampling rates for the two sets of data. The on-board data is recorded at a frequency of 1kHz while the frequency for recording Optitrack data is 120Hz. Since the lowest possible frequency for Betaflight is 500Hz and the highest capture rate for Optitrack is 360Hz, it is not possible to directly obtain the two datasets at the same frequency.

Both these datasets measure the attitude angles and therefore, the roll angle for both is compared. While Betaflight begins recording the data on arming the quadrotor, the Optitrack logs have to be manually triggered. This results in a time lag in one of the datasets which must be compensated for, before resampling. The arm and disarm times for Betaflight are denoted by t_{inBF} and t_{endBF}. Since Optitrack datasets start recording earlier and stops later, the values t_{inOT} and t_{endOT} denote the arm and disarm values on the Optitrack time range. These two values are unknown and must be computed to compensate for the time lag. Figure 3a depicts the aforementioned time values in order to have a better understanding of this process.

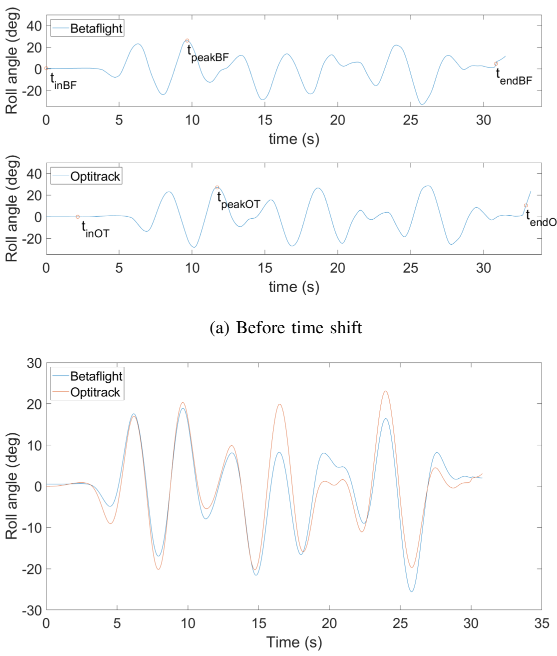
A peak value for roll angle in both datasets is identified. The corresponding time values t_{peakBF} and t_{peakOT} are noted. The time difference from the arm and disarm points to the peak value on Betaflight is noted. With the safe assumption that the time scale is the same for both the datasets, it can be said that this time difference would also be the same. With two unknown variables and Equation 12, the arm and disarm points on Optitrack are computed and the time lag is accounted for.

$$\begin{gathered} t_{peakOT} - t_{inOT} = t_{peakBF} - t_{inBF} \\ t_{endOT} - t_{peakOT} = t_{endBF} - t_{peakBF} \end{gathered} \tag{12}$$

The RMSE for the roll angle after synchronisation is 1.592 degrees. This can be attributed to the measurement error between the two sensors.

C. *Smoothing Noisy Data*

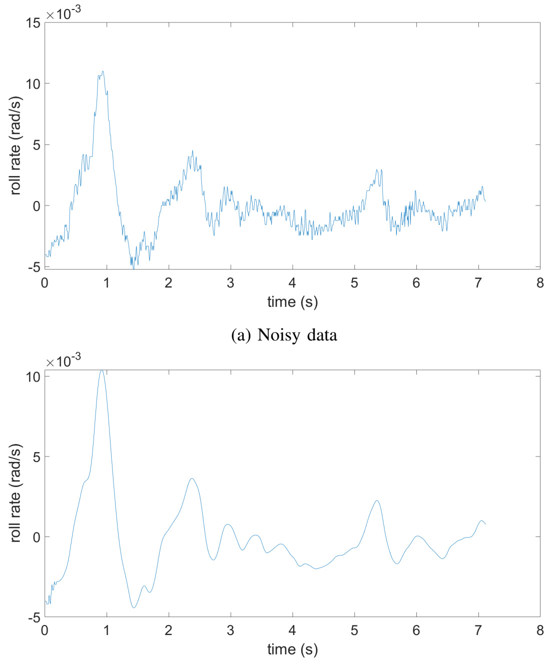
Optitrack only measures the attitude and position coordinates. Computing the linear and rotational velocities and accelerations would involve computing the derivatives of the Optitrack data. This would amplify the noise by a factor of $(\frac{1}{\Delta t})^n$, where n is the order of the derivative [11], and therefore



(a) Before time shift

(b) After time shift

Fig. 3: Roll angle measurements plotted against time for Betaflight and Optitrack, before and after synchronisation



(a) Noisy data

(b) Smooth data

Fig. 4: Noisy data obtained directly from the gyroscope smoothened by computing a moving average in a fixed window

the data must be smoothened before it is used to calculate various parameters.

Betaflight allows for an internal calculation and compensation of sensor biases if the drone is left still for a few seconds after powering it up. While it corrects for the sensor bias, it is unable to filter out the noise due to vibrations. Therefore, the data obtained from the IMUs must be rid of this noise before they are used to compute the estimations. Since the estimations use dead-reckoning, the inclusion of sensor noise would add to the errors during integration and cause higher drifts in the position estimations.

A method used to smooth both datasets is to use a low-pass moving average filter using a window of fixed length. Figure 4 depicts this method implemented on the roll rate obtained from the gyroscope. If the window length selected is too large, it could result in lower peak values and in turn, a loss of data. However, if the window length is too small, it would not be able to filter out the noise. Therefore, the window length must be selected appropriately for each dataset. In order to incorporate this method online, the window was selected with only past values included in the calculation of the average value. The window in Figure 4 is selected to be 0.2s or 25 values, therefore a small noise is seen in the beginning as the filter does not work until the 25^{th} value is obtained.

IV. System Identification

To estimate the behaviour of the quadrotor, it is important to map the control inputs and the current states of the drone to its future states. Hence, the first subsystem identified in this paper is the rate controller that takes the control inputs provided by the pilot, and provides the motor speeds as the output. The next step involves modelling the dynamics. For this step, the drag and thrust coefficients described in section II must be identified first.

A. Identification of the Rate Controller

Being a rate controller, the control inputs are commands for the rotation rates and thrust (throttle command). A schematic representation of the input-motor system is depicted in Figure 5. Since it is a closed loop control system, Betaflight uses a feedback for the gyroscope measurements to obtain the error in the rotation rates. The errors are input to a PID controller. The throttle input T_r is recorded as a percentage value (between 0 and 1). From hereon, the throttle value and the scaled output of the PID block are sent to the mixer block.

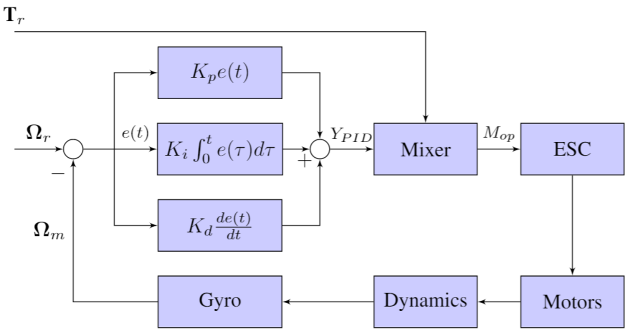


Fig. 5: Rate control block diagram for a quadrotor using Betaflight

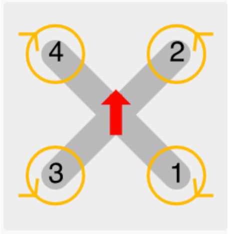


Fig. 6: Direction of spin for each motor of the quadrotor

The mixer values are generally set between 0.0 and 1.0 for throttle and -1.0 to +1.0 for the three rate inputs. The mixer value indicates how much authority the motor has to each specific input and the sign indicates the direction. For example, +1.0 for yaw rate indicates a counterclockwise rotation while -1.0 indicates a clockwise rotation. Figure 6 depicts the number and direction of spin for each motor, which determine the signs for each input used in the mixer. The mixer used in this experiment is depicted in Table I where the row corresponds to the mixer values for throttle, roll rate, pitch rate, and yaw rate respectively, of the i^{th} motor.

Motor	T	R	P	Y
m_1	1.0	-1.0	1.0	-1.0
m_2	1.0	-1.0	-1.0	1.0
m_3	1.0	1.0	1.0	1.0
m_4	1.0	1.0	-1.0	-1.0

TABLE I: Motor mixer M for the quadrotor

The equation to obtain the rate mix value of each motor is:

$$M_i = \frac{Y_r * M_{r_i} + Y_p * M_{p_i} + Y_y * M_{y_i}}{1000} \tag{13}$$

where M_i is the total rate value for the i^{th} motor, Y represents the PID output along a specific axis in deg/s, M_{axis_i} represents the mixer value along the axis for the i^{th} motor from Table I, and r, p, y stand for roll axis, pitch axis, and yaw axis respectively. M_i is usually a value between -1.0 and 1.0.

Betaflight has a setting known as airmode which checks for motor saturation commands and alters the throttle setpoint if required. This is to ensure that if the input command for the motor exceeds 100%, higher preference is given to the execution of rate commands over throttle commands. If the throttle input is over 50%, the throttle setpoint is restricted. From 4 rate mix values (for each motor of a quadrotor), the throttle is constrained in the interval $[-min(M_i), 1 - max(M_i)]$. For example, if the throttle setpoint is 0.8 and mix values of the motors are 0.2, 0.3, -0.5, -0.1, the throttle is set to $1 - 0.3 = 0.7$.

This ensures that the motor commands are neither saturated (greater than 1.0) nor under-valued (less than 0.0). Therefore the motor command for each motor can be obtained using Equation 14.

$$M_{cmd} = M_i + T_{adj} \tag{14}$$

where M_{cmd} is a vector corresponding to the motor command for each motor in order, M_i is the total rate mix for each motor, and T_{adj} is the adjusted value of the throttle.

In order to scale it to the actual motor command value, the motor protocol used to communicate commands from the flight controller to the ESCs must be known. In this case, the D-shot protocol is used.

In D-shot (digital protocol), the flight controller sends a 11-bit packet ($2^{11} = 2048$). 0 indicates the drone is disarmed, 1-47 are reserved for special commands, 48 and 2047 are the minimum and maximum commands used for control respectively. The drone can be set to have an idle speed (IS), which is the speed when the quadcopter is armed with the rest of the inputs at zero. For example, using Equation 15, an idle speed of 5.5% for D-shot would correspond to a motor command of 158 on arming. An idle speed is usually set to avoid motor desynchronisation.

$$IS = cmd_{min} + idle\% * (cmd_{max} - cmd_{min}) \tag{15}$$

With the idle speed known, the digital value to be sent to the ESCs can be obtained using Equation 16.

$$M_{op} = IS + M_{cmd} \, (cmd_{max} - IS) \tag{16}$$

While this provides the commands sent to the ESCs, it still does not provide the speed of each motor. The speed of each rotor is measured in the form of ERPM (electric RPM), that is related to the mechanical RPM using Equation 17.

$$RPM = ERPM \times np \tag{17}$$

where np stands for the number of magnet pole pairs. In the drones used in this experiment, each motor has 14 magnets and therefore $np = 7$. The range of ERPM is dependent on the voltage source. A two-cell (2S) battery would provide approximately half the ERPM when compared to a four-cell (4S) battery for the same quadrotor setup. Figure 7 shows the plot of the motor commands and ERPM for the Eachine Wizard drone using a 4S battery. Assuming a linear relationship between the motor commands and ERPM (and correspondingly the RPM), linear regression is used to compute the correlation between the two terms. The coefficient is found to be $k_{rpm} = 2.23$ with an R^2 fit is 0.867 for the Eachine drone and $k_{rpm} = 0.467$ with an R^2 fit is 0.915 for the Diatone drone.

Using the above equations, the RPM of the drone is estimated for various flight experiments. Figure 9 depicts the estimated RPM to the measured RPM values for a flight without any saturation. The RMSE for several flights for each motor is depicted in Table II.

B. Identification of the Drag Coefficients

The drag coefficients in the x and y direction in the body frame are identified using the accelerometer data obtained from Betaflight, and the velocity data obtained from Optitrack. With the assumptions that there is no wind in the indoor flying arena, and the accelerometer bias has already been corrected

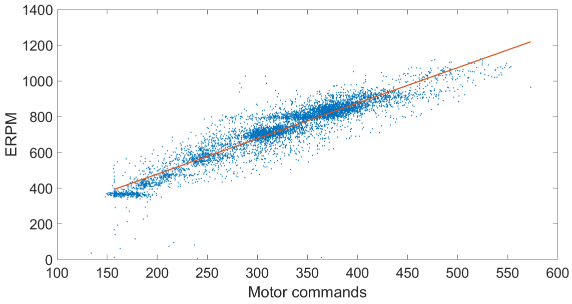


Fig. 7: Linear regression used to compute the ERPM coefficient for the Eachine Wizard drone (4S battery)

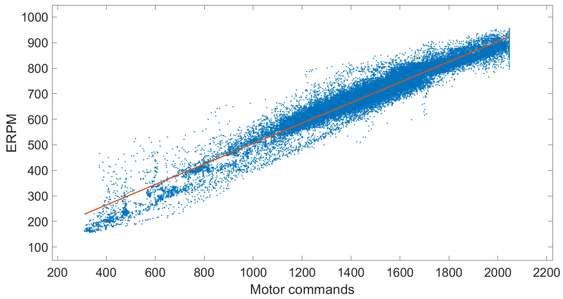


Fig. 8: Linear regression used to compute the ERPM coefficient for the Diatone drone (2S battery)

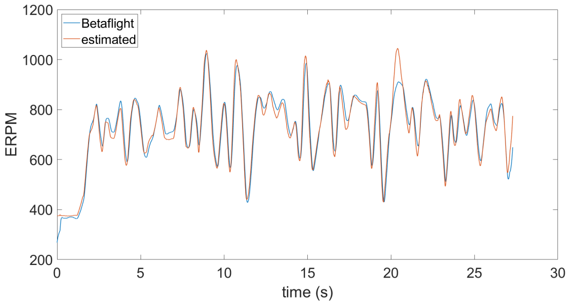
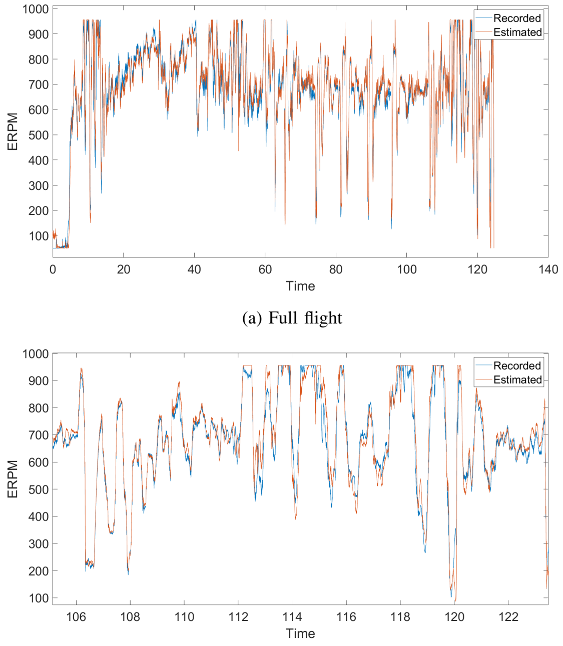


Fig. 9: Estimated ERPM vs Measured ERPM for a flight with no saturations

Flight	Dataset	Motor 1	Motor 2	Motor 3	Motor 4
No Saturation	1	4.0067	3.5311	1.0434	1.0781
	2	2.5957	5.9795	2.9641	2.5881
	3	3.7711	4.5426	5.3978	1.2587
	4	7.2332	1.5678	2.1929	3.0357
Saturation	5	10.7754	12.5235	11.9000	12.8743
	6	5.8668	5.0154	10.1764	9.6069
	7	7.4061	8.6456	7.5253	8.2602

TABLE II: RMSE for the estimated RPM of each motor for four different flight tests



(a) Full flight

(b) Zoomed in to show a part where the motor reaches saturations

Fig. 10: Estimated ERPM vs Measured ERPM for a flight reaching motor saturations

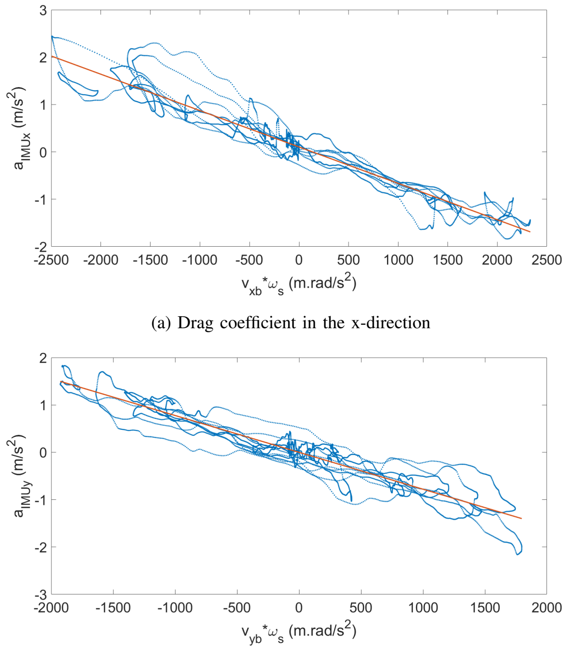
for by Betaflight, a linear relationship between the specific acceleration (sum of the quadcopter accelerations and the acceleration due to gravity in the body frame) measured by the IMU and the product of velocity in the body frame and motor speed can be obtained. To obtain the velocity values in the body frame, the product of the rotation matrix (Equation 3) and derivative of the position coordinates is computed.

$$\begin{bmatrix} a_{IMUx} \\ a_{IMUy} \end{bmatrix} = \begin{bmatrix} k_{dx} & 0 \\ 0 & k_{dy} \end{bmatrix} \begin{bmatrix} v_{bx} \\ v_{by} \end{bmatrix} \omega_s \tag{18}$$

Two flight experiments are performed; one forward flight and one lateral flight. This is done to maximise excitation in the body frame x and y-directions to obtain more accurate drag coefficients. Figure 11 depicts the plot of the IMU measurements against the product of velocity and the average speed of the four motors (in rad/s). The R^2 fit for the two figures is found to be 0.889 and 0.881 respectively. The coefficients computed are $k_{dx} = -7.674 \times 10^{-4} rad^{-1}$ and $k_{dy} = -7.78 \times 10^{-4} rad^{-1}$

C. Identification of the Thrust Coefficients

The first step taken in this case is recording a dataset with the drone hovering over the ground. At near-hover conditions, it can be assumed that the body velocities do not have an effect on the thrust generated. For the standard thrust model, thrust is assumed to be a linear function of the square of the motor speed. The IMU measurements in the z-direction are plotted against the sum of the squared motor speeds as depicted in Figure 12. Using least squares linear regression,



(a) Drag coefficient in the x-direction

(b) Drag coefficient in the y-direction

Fig. 11: Linear regression used to compute the drag coefficient

the value of the coefficient K_t in Equation 5 is found to be $5.23 \times 10^{-6} Ns^2/rad^2$, with an R^2 fit of 0.718.

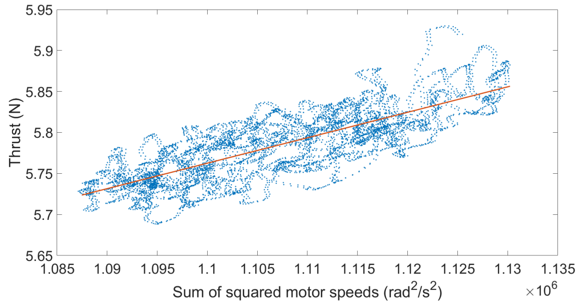


Fig. 12: Linear regression used to compute the coefficient relating the square of motor speeds to thrust

To include the effect of the body velocities and induced velocity, it is important to excite the system appropriately. Therefore, a forward flight dataset is used to calculate the thrust coefficients. It is important to note that the previously calculated value of k_t cannot be used in this case. Looking back at Equation 9, and summing it up for all 4 propellers, with the assumptions that vertical velocity $V_{z_i} = V_z$ and horizontal velocity $V_{h_i} = V_h$,

$$\mathbf{T} = \sum_{i=1}^{4} T_i = k_t \sum_{i=1}^{4} \omega_i^2 - k_z V_z \sum_{i=1}^{4} \omega_i + k_h V_h^2 \tag{19}$$

where $V_h = \sqrt{V_x^2 + V_y^2}$ Therefore, the coefficients to be computed are k_t, k_z, and k_h.

Since there are 4 parameters in this equation, it is difficult to depict the variation of thrust with the other 3 parameters. The coefficients are computed using multi-variable linear regression. The computed coefficients are $k_t = 5.22 \times 10^{-6} Ns^2/rad^2$, $k_z = 9.12 \times 10^{-4} Ns^2/m.rad$, $k_h = 2.23 \times -3 Ns^2/m^2$, with an R^2 fit of 0.953. Figure 13 depicts the thrust estimated using these coefficients on a validation dataset with a fit of 0.914.

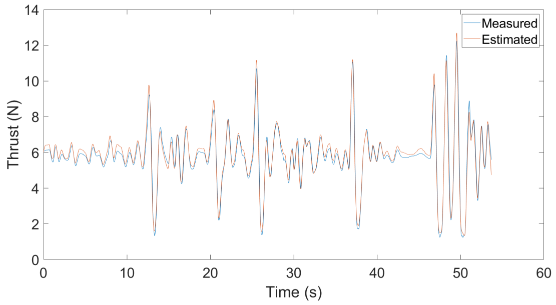


Fig. 13: Comparison of estimated and measured thrust for a vertical flight dataset

V. Results

The model is complete with the computed drag and thrust coefficients. Therefore, validation datasets are to be created. Different types of maneuvers are attempted to understand how reliable the estimation models are and how long can the estimations have lower errors before they have large drifts from the groundtruth.

A. *Maneuvers*

Two different types of maneuvers are used to follow three different shaped paths to create validation datasets for the estimation model. The three different paths are a L-shaped path, a U-shaped path and a square shaped path, thereby increasing the length of the dataset with each attempt. The first maneuver uses maximum pitch and roll movements, and as less yaw movements as possible to follow the path. The second maneuver also includes yaw so that the drone always faces the direction in which it is moving. This is done with the idea that the inclusion of yaw inputs decreases the speed with which a quadrotor can follow the same path due to slower rotations.

B. *Estimation Models used for comparisons*

The results for the model are compared to the groundtruth but also to three other estimation models. The first model makes use of the accelerometer readings to compute the body frame accelerations [12]. Drag is compensated for in the x- and y-directions, with no compensation in the z-direction. This is referred to as the “Acceleration-based model”.

The second model is one that uses the standard thrust model (Equation 5) [6]. This model compensates for drag along the x- and y-axes. However, it has no compensation in the z-axis, neither with drag nor with the thrust. It also does not take the Coriolis accelerations into account during the acceleration computations. Therefore, the only difference between the first two models is the use of standard thrust to compute accelerations as opposed to accelerometer readings. This is referred to as the “Base model” in this section.

The third model uses the refined thrust model (Equation 9) to determine the accelerations [13]. This model does not include the effects due to the Coriolis accelerations. It is referred to as the “Thrust Compensated model”. The proposed estimation model is referred to as the “Coriolis compensated model”. As the name suggests, it accounts for Coriolis effect. This is the only difference between the Thrust compensated model and the Coriolis compensated model.

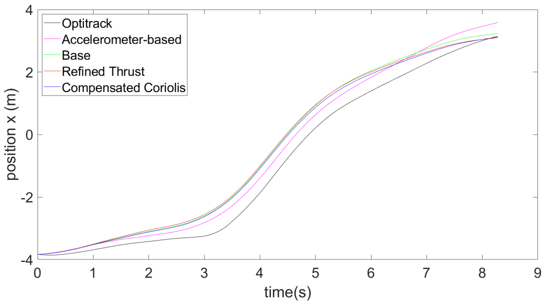


Fig. 14: Groundtruth positions (x-axis) compared to the accelerometer-based estimation, base model, refined thrust model, and Coriolis compensated model

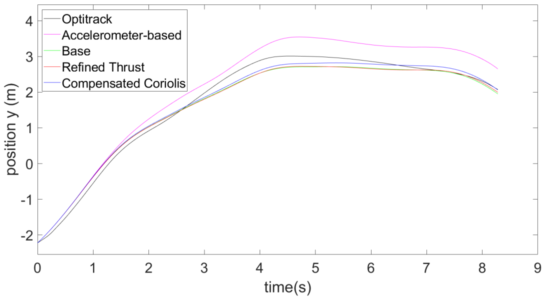


Fig. 15: Groundtruth positions (y-axis) compared to the accelerometer-based estimation, base model, refined thrust model, and Coriolis compensated model

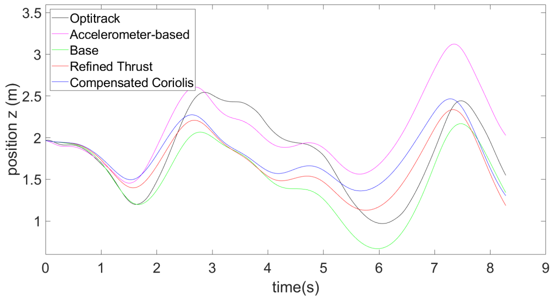


Fig. 16: Groundtruth positions (z-axis) compared to the accelerometer-based estimation, base model, refined thrust model, and Coriolis compensated model

C. *Comparative results*

The Root Mean Square Errors (RMSE) for the position estimates are represented in Table III.

Maneuver	Method	X	Y	Z
Yaw L-shape	Accelerometer-based	0.4631	0.3947	0.2942
	Base	0.4717	0.0975	0.2725
	Refined Thrust	0.4137	0.0671	0.0980
	Coriolis	0.3589	0.0195	0.0263

TABLE III: Comparison of the RMS position errors for different methods

The results for the remaining maneuvers are depicted in section A.

VI. Conclusions

The current smoothing filter might be computationally expensive as it requires a set of past values and the present value in order to obtain the smoothened data. Using such an algorithm on-board during a race might prove to slow the controller down and a more computationally viable filter must be implemented.

The model for estimating the rate controller is quite reliable with a fit of over 0.9 in 7 datasets, 4 with slower flights and the motors at about 30% of their capacity, and 3 with saturations, each at least 30 seconds long. The two kinds of datasets also represent two separate drones and therefore shows that this model is robust for different quadrotor designs as long as they are working with the Betaflight controller firmware.

From the position estimation results, it is seen that thrust compensation indeed improves the estimations vastly in the x- and z-directions. Compensating for the Coriolis acceleration further reduces the estimation errors by about 5cm. The model also has some shortcomings. It has not yet been verified for aggressive flights where the physical limits of the motors are tested. This is further explained in section A.

The model estimations for the position coordinates beginning from take-off are extremely erroneous. The model estimations are comparable when the estimation is started from a near-hover position, which is not possible during a drone race. The use of dead-reckoning implies that the estimations have errors lower than 1m for datasets that are less than 15 seconds. For longer datasets, the drifts add up and the trajectory generated by the estimator is no longer similar to the groundtruth.

The estimations are also dependent on the gyroscope measurements for the attitudes and their rates. The biases are assumed to be negligible in the case of attitude measurements, which could lead to certain drifts in the long run. The position estimations could improve further if the attitude measurements could be substituted with more accurately estimated attitudes.

These issues must be compensated for estimating the states of a racing drone because the races could be longer, and may not provide the groundtruth to be able to find an initialisation point.

References

[1] Moon, H., Martinez-Carranza, J., Cieslewski, T., Faessler, M., Falanga, D., Simovic, A., Scaramuzza, D., Li, S., Ozo, M., De Wagter, C. and de Croon, G., 2019. “Challenges and implemented technologies used in autonomous drone racing.” *Intelligent Service Robotics*, 12(2), pp.137-148.

[2] Das, A., Subbarao, K. and Lewis, F., 2009. “Dynamic inversion with zero-dynamics stabilisation for quadrotor control.” *IET control theory & applications*, 3(3), pp.303-314.

[3] Pounds, P., Mahony, R. and Corke, P., 2010. “Modelling and control of a large quadrotor robot.” *Control Engineering Practice*, 18(7), pp.691-699.

[4] Gremillion, G. and Humbert, J., 2010, August. “System identification of a quadrotor micro air vehicle.” In *AIAA Atmospheric Flight Mechanics Conference* (p. 7644).

[5] Leishman, R.C., Macdonald, J.C., Beard, R.W. and McLain, T.W., 2014. “Quadrotors and accelerometers: State estimation with an improved dynamic model.” *IEEE Control Systems Magazine*, 34(1), pp.28-41.

[6] Leishman, G.J., 2006. *Principles of helicopter aerodynamics with CD extra.* Cambridge university press.

[7] Mahony, R., Kumar, V. and Corke, P., 2012. “Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor.” *IEEE Robotics and Automation magazine*, 19(3), pp.20-32.

[8] Houghton, E.L. and Carpenter, P.W., 2013. *Aerodynamics for engineering students.* Elsevier.

[9] Bangura, M., Melega, M., Naldi, R. and Mahony, R., 2016. “Aerodynamics of rotor blades for quadrotors.” *arXiv preprint arXiv:1601.00733.*

[10] Bangura, M. and Mahony, R., 2012. “Nonlinear dynamic modeling for high performance control of a quadrotor.” *Proceedings Australasian Conference on Robotics and Automation 2012*

[11] Caetano, J.V., Percin, M.U., van Oudheusden, B.W., Remes, B., De Wagter, C., de Croon, G.C.H.E. and de Visser, C.C., 2015. “Error analysis and assessment of unsteady forces acting on a flapping wing micro air vehicle: free flight versus wind-tunnel experimental methods.” *Bioinspiration & biomimetics*, 10(5), p.056004.

[12] Li, S., Ozo, M.M., De Wagter, C. and de Croon, G.C., 2020. “Autonomous drone race: A computationally efficient vision-based navigation and control strategy.” *Robotics and Autonomous Systems*, 133, p.103621.

[13] Svacha, J., Mohta, K. and Kumar, V., 2017, June. “Improving quadrotor trajectory tracking by compensating for aerodynamic effects.” In *2017 international conference on unmanned aircraft systems (ICUAS)* (pp. 860-866). IEEE.

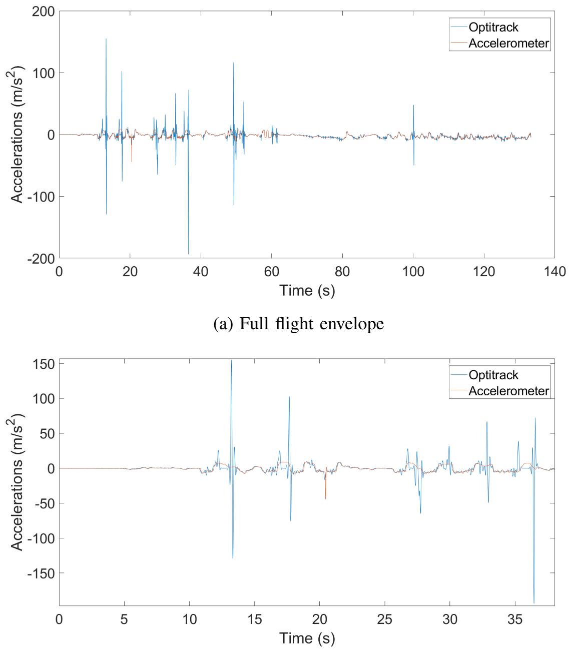
Appendix A
Results and Tables

The appendix explains why the position estimator was not verified for motor saturations. It also depicts the results for various maneuvers described in section V, with the quadrotor flying at lower speeds.

A. *Estimation during Motor Saturations*

To estimate the position, it is important to correctly estimate the accelerations. For this, the thrust and drag parameters must be identified with a certain confidence. Being a different drone than the one used to estimate the results depicted in section V, the thrust and drag parameters could not be the same. To obtain the thrust parameters, the reference used was the accelerometer measurements. However, a comparison of the accelerometer measurements with the measurements obtained from Optitrack showed an interesting case.

Figure 17 depicts that the double differentiation of position measurements has varying peaks, some of which are even up to 180 m/s^2, seemingly unlikely in an area such as the Cyberzoo. On the other hand, the accelerometer measurements are so heavily filtered for this particular drone that none of the expected peaks are depicted in this figure (a very smooth plot for a quadrotor flying aggressively).



(a) Full flight envelope

(b) Zoomed in to depict a portion of the flight involving more peaks

Fig. 17: Comparison of acceleration measurements along the z-axis obtained from Optitrack and Betaflight (accelerometer readings)

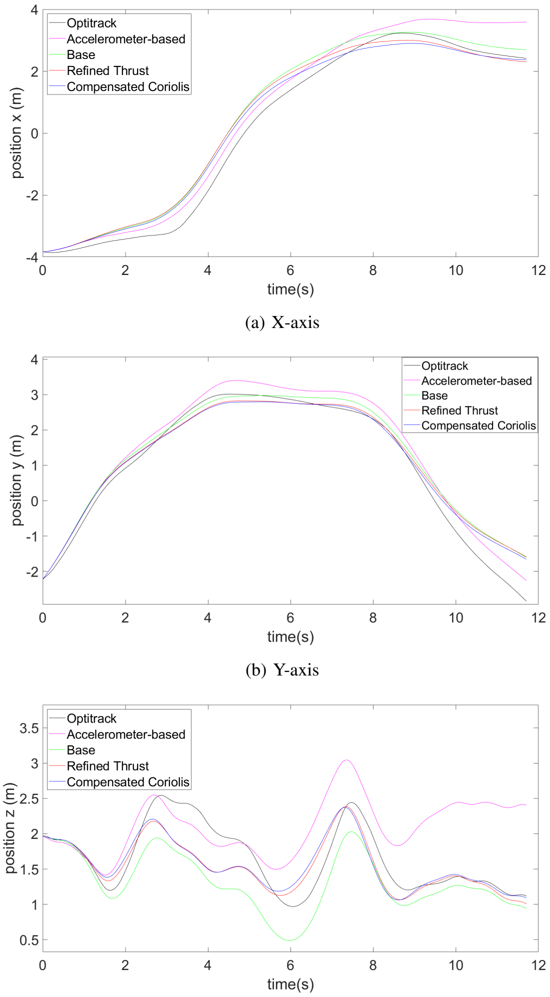
This makes it difficult to obtain the required parameters for the position estimation model. The position estimation model has not been verified for motor saturations due to insufficient/erroneous data.

B. *Position Estimation Results*

The results have been tabulated in the form of RMSE of the errors.

Maneuver	Method	X	Y	Z
Yaw U-shape	Accelerometer-based	0.4859	0.1764	0.4876
	Base	0.4311	0.2057	0.0953
	Refined Thrust	0.3650	0.1442	0.0643
	Coriolis	0.1858	0.0761	0.0391
Yaw Square shape	Accelerometer-based	0.5610	0.1151	0.7810
	Base	0.5160	0.2452	0.1198
	Refined Thrust	0.4557	0.1901	0.0424
	Coriolis	0.2720	0.1523	0.0144
No yaw L-shape	Accelerometer-based	0.3258	0.1756	0.2792
	Base	0.2555	0.1366	0.2053
	Refined Thrust	0.2252	0.1057	0.0653
	Coriolis	0.2094	0.0766	0.0087
No yaw U-shape	Accelerometer-based	0.2698	0.2131	0.4348
	Base	0.2098	0.1206	0.2953
	Refined Thrust	0.1907	0.1007	0.0828
	Coriolis	0.1817	0.0895	0.0136

TABLE IV: Comparison of the RMS position errors for different methods

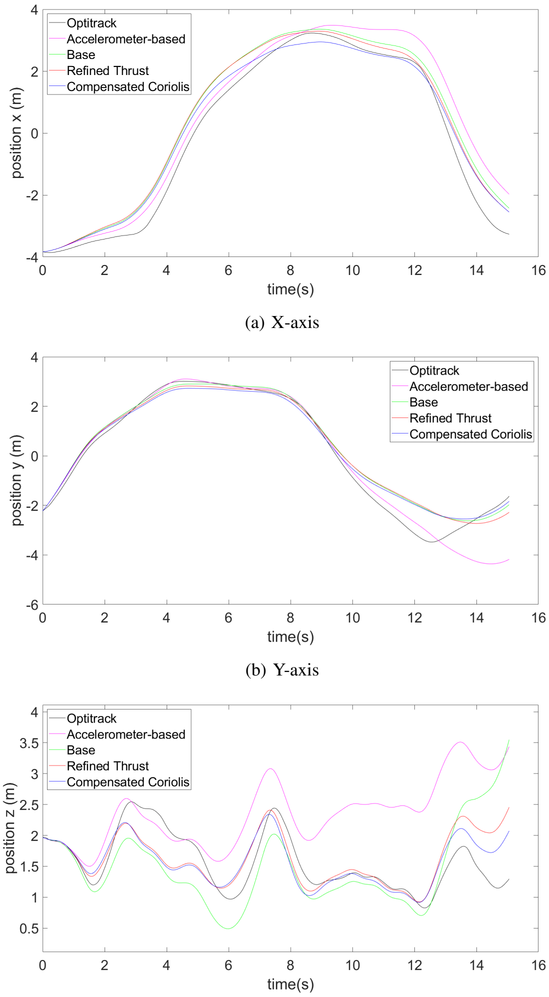


(a) X-axis

(b) Y-axis

(c) Z-axis

Fig. 18: Different estimation models compared to the groundtruth (Yaw maneuver for a U-shaped track)

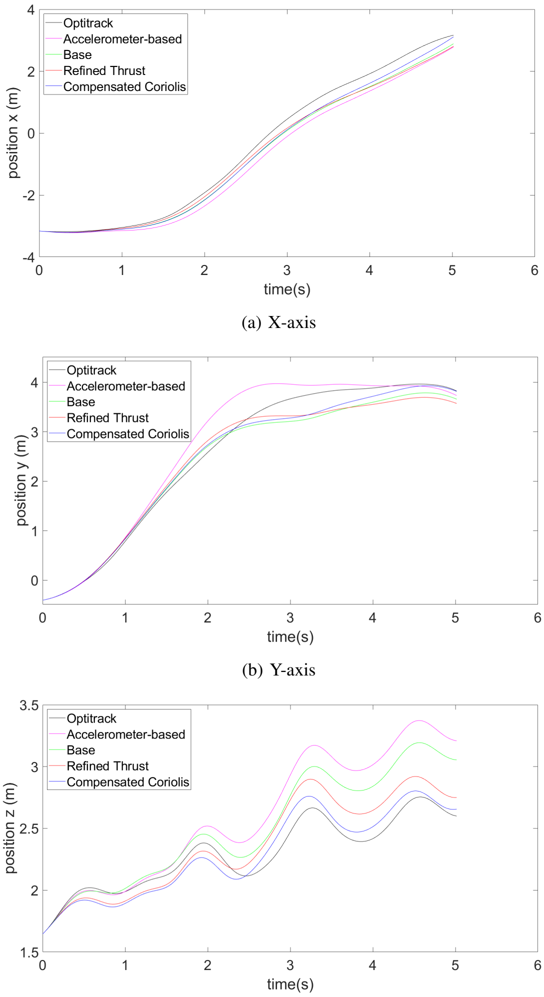


(a) X-axis

(b) Y-axis

(c) Z-axis

Fig. 19: Different estimation models compared to the groundtruth (Yaw maneuver for a square-shaped track)

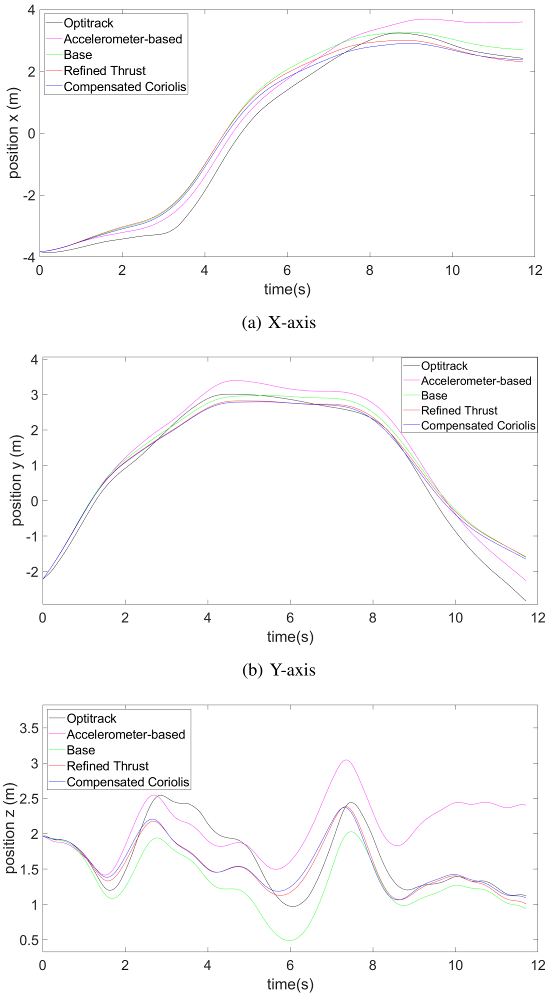


(a) X-axis

(b) Y-axis

(c) Z-axis

Fig. 20: Different estimation models compared to the groundtruth (No yaw maneuver for a L-shaped track)



(a) X-axis

(b) Y-axis

(c) Z-axis

Fig. 21: Different estimation models compared to the groundtruth (No yaw maneuver for a U-shaped track)

Part II

Literature Study

Chapter 2

Quadrotor Modelling

Drone racing involves quadrotors flying at high speeds. If not performed in a right manner, this poses a risk to the drone itself as well as the people and property in the vicinity of the testing area. Therefore, it is very important to verify the control strategy of a drone using simulations before testing it in the real world.

Mathematical models of a quadrotor can be designed in two different ways - using the theoretical principles of dynamics and aerodynamics to model the equations of motion, or using system identification methods on real flight data (chapter 3). System identification makes use of statistical methods to provide a relationship between the inputs and the outputs, and may not require an understanding of the working of the system. Modelling the system using equations requires an understanding of its working.

2-1 Dynamic modelling

The quadrotor is modelled as a rigid body with under-actuated dynamics as four inputs are used to control six DOF (three translational and three rotational). A dynamic model helps describe the motion of the quadrotor in response to the actions of the propellers, and aerodynamic effects such as wind and drag. The simplest model of a quadrotor represents the UAV as a rigid body with inertia. Aerodynamic effects are neglected and gravity is the only external force acting upon the body (Pounds et al. [2010]).

To begin modelling the quadrotor dynamics, the inertial and body reference frames E_i and E_b are first defined (Wang et al. [2016]). The position vector is described by the vector $\boldsymbol{\xi} = \begin{bmatrix} x & y & z \end{bmatrix}^T \in E_i$. The orientation vector is described as $\boldsymbol{\eta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \in E_i$. The linear and angular velocity vectors are defined in the body frame as $\boldsymbol{V} = \begin{bmatrix} u & v & w \end{bmatrix}^T \in E_b$ and $\boldsymbol{\Omega} = \begin{bmatrix} p & q & r \end{bmatrix}^T \in E_b$. The relations between these parameters is depicted in Equation 2-1, where $\mathbf{R}$ and $\mathbf{J}$ are the rotational and angular velocity transformation matrices respectively.

$$\begin{aligned} \dot{\boldsymbol{\xi}} &= \mathbf{RV} \\ \dot{\boldsymbol{\eta}} &= \mathbf{J\Omega} \end{aligned} \tag{2-1}$$

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & -sin\theta \\ 0 & cos\phi & sin\phi cos\theta \\ 0 & -sin\phi & cos\phi cos\theta \end{bmatrix} \tag{2-2}$$

$$\mathbf{R} = \begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - cos\phi sin\psi & cos\phi sin\theta cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi \\ -sin\theta & cos\phi sin\theta & cos\phi cos\theta \end{bmatrix} \tag{2-3}$$

Different mathematical techniques can be used to derive the dynamic equations of a quadrotor. Two common methods include using the Newton-Euler method and the Euler-Lagrange method. The Newton-Euler method focuses on the rigid body equations based on Newton's second law of motion (Bolandi et al. [2013], Deif et al. [2014]). The Euler-Lagrange method uses kinematics and energy equations to obtain the mathematical model (Bouabdallah [2007], Carrillo et al. [2012]). Ideally, the same mathematical model must be obtained using either technique. The equations obtained can be represented in the state-space form as depicted in Equation 2-4, where **X** is the state vector of the system, **U** is the input vector and **Y**, the output vector of the system.

$$\begin{aligned} \dot{\mathbf{X}} &= A(\mathbf{X}) + B(\mathbf{X}, \mathbf{U}) \\ \mathbf{Y} &= C(\mathbf{X}, \mathbf{U}) \end{aligned} \tag{2-4}$$

The states of the quadrotor system are usually the vectors for the position, the translational velocities, the attitude (Euler) angles and the rates of these angles. The following equation describes the dynamics of each rotor with the assumption that aerodynamic effects such as blade flapping and ground effect are neglected.

$$\begin{aligned} f_i &= k_f \omega_i^2 \\ \tau_i &= k_m \omega_i^2 \end{aligned} \tag{2-5}$$

where k_f and k_m are the force and moment constants and ω_i is the speed of the i^{th} motor.

The inputs are the total thrust and the torques generated, given by Equation 2-6, where l is the length of the quadrotor arm, from the center of mass to the center of one rotor.

$$\begin{aligned} u_1 = T &= k_f \sum_{i-1}^{4} w_i^2 \\ u_2 = \tau_\phi &= lk_f[(\omega_2^2 + \omega_3^2) - (\omega_1^2 + \omega_4^2)] \\ u_3 = \tau_\theta &= lk_f[(\omega_1^2 + \omega_2^2) - (\omega_3^2 + \omega_4^2)] \\ u_4 = \tau_\psi &= k_m[(\omega_1^2 + \omega_3^2) - (\omega_2^2 + \omega_4^2)] \end{aligned} \tag{2-6}$$

Assuming a symmetrical quadrotor, the equation of motion is described in Equation 2-7, where **I** is a diagonal matrix of the inertia along the three axes.

$$\mathbf{I}\ddot{\eta} \times \mathbf{I}\dot{\eta} = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{2-7}$$

On simplifying Equation 2-7, we get the non-linear state space equation for the rates of the euler angles (Equation 2-8).

$$
\begin{aligned}
\ddot{\phi} &= \frac{u_2}{I_{xx}} + \dot{\theta}\dot{\psi}\frac{(I_{yy} - I_{zz})}{I_{xx}} \\
\ddot{\theta} &= \frac{u_3}{I_{yy}} + \dot{\phi}\dot{\psi}\frac{(I_{zz} - I_{xx})}{I_{yy}} \\
\ddot{\psi} &= \frac{u_4}{I_{zz}} + \dot{\phi}\dot{\theta}\frac{(I_{xx} - I_{yy})}{I_{zz}}
\end{aligned}
\tag{2-8}
$$

Newton's second law of motion provides the equation for the acceleration for translational motion of the quadrotor.

$$
m\ddot{x} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \mathbf{R}F_b \tag{2-9}
$$

where F_b is the sum of the forces acting on the quadrotor

$$
F_b = \begin{bmatrix} 0 \\ 0 \\ -u_1 \end{bmatrix} \tag{2-10}
$$

Solving equations 2-9 and 2-10, the accelerations are obtained as seen in Equation 2-11.

$$
\begin{aligned}
\ddot{x} &= -\frac{u_1}{m}(cos\phi sin\theta cos\psi + sin\phi sin\psi) \\
\ddot{y} &= -\frac{u_1}{m}(cos\phi sin\theta sin\psi - sin\phi cos\psi) \\
\ddot{z} &= g - \frac{u_1}{m}(cos\phi cos\theta)
\end{aligned}
\tag{2-11}
$$

2-2 Linearisation

The model obtained is non-linear, making it difficult to transform the model from the time domain to the frequency domain. This also makes it difficult to implement classical control techniques like PID control (Musa [2018]). Linearisation is performed using the first-order Taylor series approximation around a point of interest, usually selected as the point of hover (Wang et al. [2016], Gonzalez-Sanchez et al. [2013]). The most common point of interest is the hover state because the pitch and roll angles are maintained at zero degrees (allowing the use of small angle approximations). This state also maintains its altitude and has limited translational movement, since translation can only occur with a change in the pitch and roll angle.

The simplified equations for the translational and rotational accelerations are depicted in Equation 2-12

$$\begin{aligned}
\ddot{x} &= -g\theta \\
\ddot{y} &= g\phi \\
\ddot{z} &= g - \frac{u_1}{m} \\
\ddot{\phi} &= \frac{u_2}{I_{xx}} \\
\ddot{\theta} &= \frac{u_3}{I_{yy}} \\
\ddot{\psi} &= \frac{u_4}{I_{zz}}
\end{aligned} \tag{2-12}$$

Three other equilibrium points can be used for linearising the dynamic model; vertical movements at a fixed velocity, and horizontal movements at a fixed pitch or a fixed roll angle (Zhang et al. [2014]).

While linearisation simplifies the model dynamics, it eliminates the gyroscopic effects as well as effects of drag on a drone, even if they were incorporated in the initial non-linear dynamics Musa [2018].

2-3 Aerodynamic Effects

In order to model the rotor with more theoretical accuracy, some studies also take the different aerodynamic effects such as drag forces, rotor blade flapping, thrust variation, and ground effect into account.

Drag Forces

Initial attempts at including drag forces in the dynamics considered it as a damping force opposing translational and rotational motion. Therefore the damping force was considered to be proportional to velocity and quite small (Derafa et al. [2006]). While this is enough to develop models for steady-state forward flight conditions, it is not accurate enough to predict the dynamic response for an actual drone. Further studies derived models taking into account the various kinds of drag forces affecting a quadrotor (Bristeau et al. [2009], Bangura et al. [2012]).

The induced drag is associated with rotor rigidity and is represented as

$$D_{induced} = K_I V_p \tag{2-13}$$

where V_p is the velocity of the rotor in the x-y plane

The translational drag is described using different equations for modelling at low speeds(Equation 2-14) and high speeds, when $V_p > \omega$ (Equation 2-15)

$$D_{translational} = K_{T_1} V_p \tag{2-14}$$

$$D_{translational} = K_{T_2}(-V_z + v_i)^4 V_p \tag{2-15}$$

The profile drag affects the transverse velocity of the rotor blades.

$$D_{profile} = K_P V_p \tag{2-16}$$

Parasitic drag is induced by the non-lifting surfaces (airframe, motors, microcontrollers) of the quadrotor.

$$D_{par} = K_{par}|V|V \tag{2-17}$$

where V is the three dimensional translational velocity vector and $K_{par} = \frac{1}{2}\rho S C_{D_{par}}$. For velocities lower than 10m/s, parasitic drag is often ignored.

Thrust variation

Analysing thrust variation for quadrotors gives us more information on two dependent effects - effective translational lift and the change in the angle of attack (AOA) leading to a change in the thrust (Huang et al. [2009]). An increase in lift is caused due to the relative momentum of the air stream when a rotor moves translationally.

The induced velocity (v_h) is related to the thrust at hover (T_h) by Equation 2-18, where ρ is the air density and A is the area covered by the rotor blades.

$$v_h = \sqrt{\frac{T_h}{2\rho A}} \tag{2-18}$$

The induced velocity in translational flight and hover are related by Equation 2-19, where α is the angle of attack of the rotor plane with respect to a free stream flow v_∞

$$v_i = \frac{v_h^2}{\sqrt{(v_\infty cos\alpha)^2 + (v_i - v_\infty sin\alpha)^2}} \tag{2-19}$$

Similar to the behaviour of aircraft wings, an increase in the AOA of the rotor with respect to the air stream causes an increase in the thrust for a given power input P.

$$T = \frac{P}{v_i - v_\infty sin\alpha} \tag{2-20}$$

For a hover state, the nominal power can be derived using Equation 2-21.

$$P_h = \frac{T_h^{3/2}}{\sqrt{2\rho A}} \tag{2-21}$$

Considering equations 2-20 and 2-21 allows for the derivation of a relationship between the nominal thrust at hover and the actual thrust produced in forward flight for a given motor command.

Blade Flapping

Blade flapping occurs due to a difference in the inflow velocities on the advancing and retreating blades (Hoffmann et al. [2007]). The advancing blade experiences higher lift due to an increase in the tip velocity and the retreating blade experiences a decrease in lift due to a reduction in the tip velocity. This induces a flapping (up and down oscillation) moment in the blades, causing roll and pitch moments as well as changing the direction of the thrust vector (Bangura et al. [2012]). Blade flapping effects are important for more accurate attitude control schemes.

Ground Effect

Ground effect is an aerodynamic effect where rotors produce more thrust per unit power when flying near a surface (Powers et al. [2013]). It is often neglected owing to the fact that this effect is negligible if the quadrotor is flying above a certain altitude (more than twice the rotor radius) (Johnson [2012]). While this effect causes the quadrotor to push away from the ground, it pulls the quadrotor towards a ceiling when it gets too close. Therefore, it plays an important role in thrust calculations and in building the altitude controller for flights performed close to a surface. The ratio of the thrust in and out of the ground effect is described in Equation 2-22. Ground effect is negligible when the ratio $\frac{r}{|z|} > 0.1$

$$\frac{T_g}{T_\infty} = \frac{1}{1 - \left(\frac{r}{4|z|}\right)^2 \left\{\frac{1}{1+\left(\frac{|V|}{v_i}\right)^2}\right\}} \tag{2-22}$$

Drone racing can be quite aggressive and autonomous drone racing requires a more precise model in order to perform more accurate control. For a high level of accuracy between simulations and reality, it would be necessary to include all the aerodynamic effects that take place. This is certainly a very complex and tedious task. Therefore, system identification is considered as an alternative method to simplify the process of simulating the input-output relationship for the MAV.

Chapter 3

System Identification in UAVs

System identification is a method of developing a mathematical model using experimental data (Morelli [2002]). The task is to represent the physical system as a dynamic model, using time-series data observations of the inputs and outputs of the real-world system. The goal is to have a model that can estimate the states (position, translational velocities, attitudes and body rates) of the drone, with minimum error when compared to the actual drone response.

Process of UAV System Identification

The overview of the system identification process is depicted in Figure 3-1. The main elements for obtaining a mathematical model using system identification principles, discussed in detail in Appendix A, are mentioned below(Hoffer et al. [2014]). These steps ensure that a balance between the model accuracy and complexity can be achieved.

- selecting the parameters (input and output signals)
- collecting the test data
- selecting a model structure
- choosing the method for system identification
- validation of the model.

Model Types

Unlike dynamic modelling, system identification principles have the ability to model the input-output relationship without any information about its physical working. This depends on the type of model to be used in the process. Different model types are classified based on the method of derivation. A white box model would be derived using the understanding of its underlying dynamics, as discussed in chapter 2. For a system with a partial or no understanding of its dynamics, the two model types are discussed.

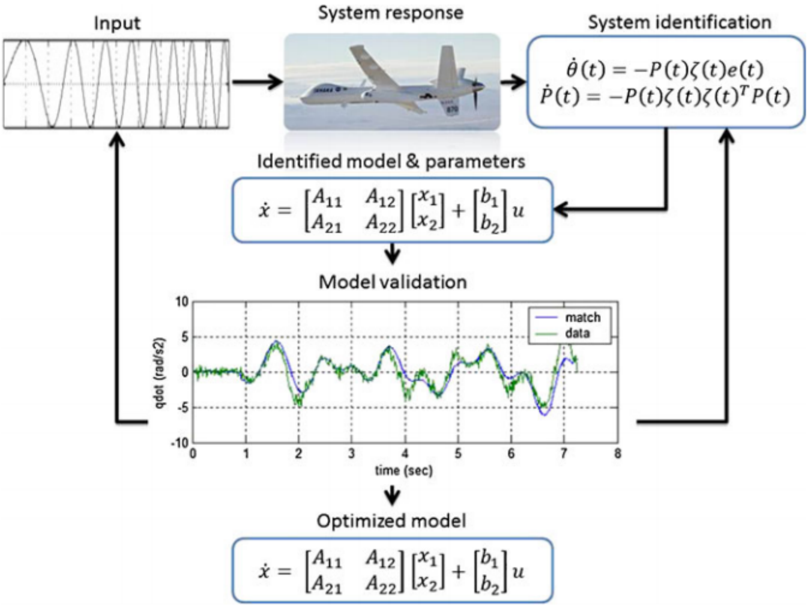

Input
System response
System identification
$\dot{\theta}(t) = -P(t)\zeta(t)e(t)$
$\dot{P}(t) = -P(t)\zeta(t)\zeta(t)^T P(t)$
Identified model & parameters
$\dot{x} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u$
Model validation
qdot (rad/s2)
time (sec)
match
data
10
5
0
-5
-10
0
1
2
3
4
5
6
7
8
Optimized model
$\dot{x} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u$

Figure 3-1: An outline of the system identification process (Hoffer et al. [2014])

Gray-Box Model

Gray-box models are somewhere in between the white-box and black-box models, i.e, they are based on a combination of physical principles and experimentally measured values. A gray-box model could be used to map the states and rotor speeds of the quadrotor to the resulting forces and moments Sun et al. [2018]. This model could further be used to predict the outputs of a controller that provides a specific set of inputs.

Black-Box Model

Black-box modelling is important when the idea is to fit data without understanding the underlying physical structure of the model (Simidjievski et al. [2020]). A simple example would be finding a transfer function for a SISO (single input single output) system. Depending on the complexity of the system, more complex methods like neural networks or multivariate splines could be used to determine the input-output relationship (Kerschen et al. [2006]). Since these models do not provide a model structure, it is difficult to verify the performance of the model.

The following chapter looks into some examples of system identification principles applied to quadrotors, with a focus on the type of model and data to be designed in this research.

Chapter 4

Examples of Quadrotor Identification

A general control loop structure for a MAV is depicted in Figure 4-1.

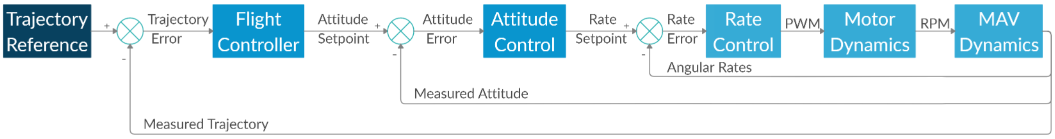


Figure 4-1: Control Loop Structure for a Micro-Aerial Vehicle

The aim of this research is to use system identification principles to model a quadrotor system that uses the Betaflight flight control software. Betaflight is used to control the rate (innermost) loop. Therefore it takes into account the actual rates of the attitudes of the quadrotor and the setpoints (or inputs provided) and returns a command (analog or digital signal) for the motors. The electronic speed controller (ESC) receives these commands and accordingly varies its voltage to induce the required RPM in the motor. The spinning motors provide the thrust and torque for the quadrotor as per its dynamics and the flight controller can record the values for the attitudes, and rates. A GPS (outdoor) or an external motion capture system (indoor) can be used to note down the values of the position, translational velocities and attitude angles of the quadrotor. This research makes use of the Optitrack motion capture system.

The idea in this study is to break down the system into several sub-systems to be identified. The first one is the system with the input as the inputs of the controller and the output as the response of the motor (or motor speed). The second sub-system uses the motor speed as inputs to obtain the dynamic states of the quadrotor (more specifically the rate of the attitude angles and the translational velocities and position in 3D). The chapter looks into the different system identification techniques used by researchers to model similar aspects. The information obtained from this chapter will be used to understand the model structures, a possible set of parameters, the identification methods used, and different experimental set-ups previously used in different studies.

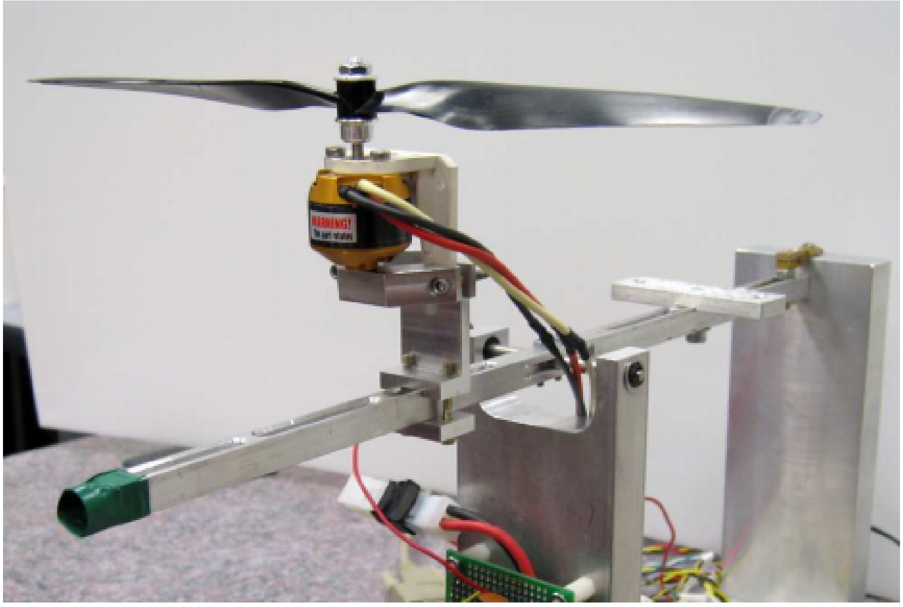


Figure 4-2: Thrust test stand (Hoffmann et al. [2007])

4-1 Motor-Rotor System Identification

The research performed by Bangura et al. [2012] focuses on making a mathematical model for a quadrotor using dynamic and aerodynamic equations. However, a small section of the research also focuses on identifying a relationship between the motor parameters and the thrust and torque coefficients using experimental data as well as some theoretical fundamentals.

For the experimental values, a thrust test stand is set up as shown in Figure 4-2. A load cell is used to measure the thrust, side force, and the torque. The voltage and current values are measured using a battery monitoring circuit (Hoffmann et al. [2007]). To measure the effects of wind, an external fan is placed and the speed of the wind is measured using a Kestral wind meter.

The equation for the mechanical power (P_m) of the motor is described in Equation 4-1. K_q and K_e are motor parameters while R_a is the electrical resistance. These three parameters are determined using linear regression, a method of parameter estimation further explained in chapter 5.

$$P_m = \frac{K_q \omega_i}{R_a}(V_a - K_e \omega_i) \tag{4-1}$$

The thrust coefficient is determined using a second order polynomial fit. The relation described in Equation 4-2 is used to obtain the torque coefficient. κ has also been determined beforehand using linear regression from the static thrust experiment.

$$C_q = \frac{C_t}{\kappa} \tag{4-2}$$

The results obtained during these tests are comparable to the results obtained during a near hover condition. However, the values determined from these thrust experiments differ when a quadrotor is performing aggressive dynamic maneuvers. To incorporate these values for a dynamic maneuver, it is important to include the effects of induced momentum due to changes in the airflow. This is an important note for the current research, as it focuses on aggressive flights rather than near-hover conditions.

Al Al et al. [2019] also proposes a study to identify the order of different relationships in the motor-rotor system. These include the relationship between the voltage source of the motor and the velocity of the rotor; the rpm of the rotor to the inflow of the rotor; the lift produced in the rotor to its velocity; and the rotor thrust to the inflow in the rotor.

The setup to obtain the experimental values is set-up as depicted in Figure 4-3. It consists of the BLDC motor, an ESC (electronic speed controller), Arduino microcontroller, and an LCD display for observing the applied voltage and the PWM value sent to the ESC from the Arduino. A slit has been made in the two plastic screens to allow for vertical movement due to the lift produced by the rotor.

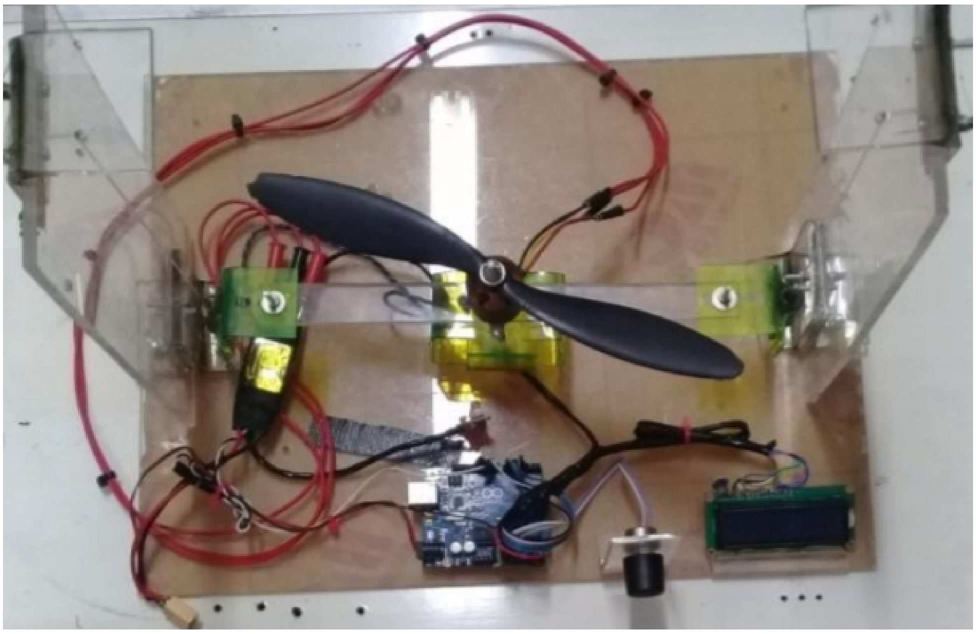


Figure 4-3: Motor-Rotor Test Stand (Al Al et al. [2019])

A fixed load is added onto the system to identify the relationship between the voltage provided and the speed of the rotor. A note is also made for the point (voltage) at which the device starts moving upwards for each specific load. The rotor lift and the change in the rotor speed are measured using the strain gauge and the tachometer respectively. An anemometer is used to measure the change in the airflow produced by the spinning rotor.

The thrust coefficient (C_T) is computed using Equation 4-3, where ρ is the air density, r is the radius of the propeller and v is the wind speed. The experiment is performed multiple times and an average value of the thrust coefficient is obtained.

$$T = \frac{C_T * \rho * \pi r^2 v^2}{2} \tag{4-3}$$

While this study takes into account the airspeed due to the induced airflow, the anemometer is fixed at the top of the stand, and therefore does not accurately measure the change in the airflow. This causes erroneous airspeed measurements and is the reason for which the thrust coefficient varies by a factor of over 0.9 in the results obtained. To understand the dynamics of the quadrotor for more aggressive maneuvers, data obtained from a high-speed flight experiment would be more suitable.

4-2 MIMO Quadrotor Identification Model

Reséndiz and Rivas-Araiza [2016] treats the quadrotor as a multi-input multi-output (MIMO) system. The inputs are the 4 PWM signals sent to the ESCs while the outputs are the euler angles as seen in Figure 4-4

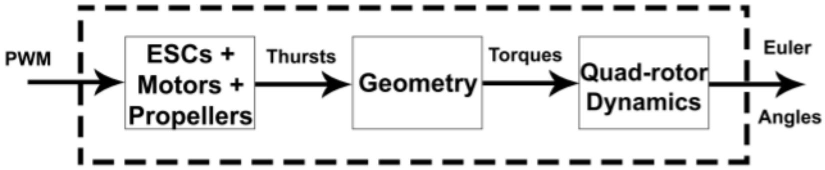


Figure 4-4: Quadrotor block diagram (Reséndiz and Rivas-Araiza [2016])

The dynamics of the quadrotor are modelled in Matlab using equations 2-6, 2-8 and 2-11. The test data used to obtain the model is represented in the time and frequency domains. The MATLAB system identification toolbox is used to formulate the transfer functions between the inputs and the outputs of the system. Since a quadrotor is a MIMO system, it is represented as multiple SISO transfer functions as depicted in Equation 4-4. $Y_1(s)$, $U_n(s)$, G_1n represent the first output, n^{th} input and the transfer function relating the first output to the n^{th} input respectively.

$$Y_1(s) = \sum_{n=1}^{4} G_{1n}(s)U_n(s) \tag{4-4}$$

The transfer functions obtained from the test data are then validated on a different dataset.

The study provides no information of the experiment performed to collect the data or the parameters measured. Since the final obtained linear model has low errors compared to the actual outputs of the non-linear quadrotor, it can be assumed that these results have been obtained for near-hover conditions. However, it provides a good basis for writing a simple mathematical model for the dynamics of a quadrotor as well as the use of the System Identification Toolbox on MATLAB. Since the toolbox can also be used to model non-linear systems in both the time and frequency domain, this study can serve as an example to obtain a non-linear model for high-speed flights.

4-3 Trajectory Alignment

While estimating the motor output from the provided input only requires the measurements of the on-board sensors and the flight controller inputs, predicting the dynamics requires more information. IMUs by themselves are prone to noise. Using the accelerometer data to calculate the positions of the quadrotor requires numerical integration. Even a slight bias in the acceleration measurements would be accumulated and over time, result in drifts in the estimated position. An external motion capture system could provide more accurate measurements of the position and orientation, and could be used to estimate the bias in the on-board sensors.

The study aims to use the Optitrack motion capture system to obtain the bias in the measurements. However the datasets obtained from Optitrack and the IMUs would have different sample rates, sample lengths, start times, and therefore would need synchronisation.

Gremillion and Humbert [2010] focuses on estimating the parameters of a linearised state-space model using a combination of the on-board sensors and the Vicon motion capture system. In order to synchronise the sample rates, a series of fiducial yaw motions are manually implemented on the quadrotor. To determine the time delay, a cross correlation is performed on the yaw rates recorded by the Vicon system and the on-board gyroscope. Once the delay is known, consistent sampling rates are obtained by linearly interpolating the time histories.

Sun et al. [2019] explores the aerodynamic effects on a quadrotor during a high-speed flight in a wind tunnel. This study also uses a combination of on-board sensors and a motion capture system (Optitrack) to model the quadrotor. In this case, the IMU bias is calculated using an EKF. A Butterworth low-pass filter is used on the unbiased IMU measurements. The power spectral densities of the accelerometer and gyroscope measurements are plotted to determine filter cut-off frequencies for reducing the noise in the measurements. The final processed data is used to calculate the forces and moments on the quadrotor.

Schubert et al. [2018] provides a novel dataset for evaluating VI odometry for SLAM applications. This study synchronises the information from a camera, a 3 axis IMU and a motion capture system. To align the IMU and motion capture system, the time shift between the two sensors must be computed.

To tackle the first problem, angular velocities computed from MoCap poses are compared to the gyroscope measurements. A coarse alignment based on the measurement time is done using a grid search algorithm. A parabola is fitted around the minimum of this search, and the minimum of this parabola is taken to be the time offset. In this process, before comparing the measurements it must be ensured that the two datasets are in the same frame of reference. While the IMU measures the accelerations and rotations in the body frame, the motion sensor measures the same in the world frame. Using the data without compensating for this would also result in incorrect bias measurements. Calculating a time shift to align the datasets is the preferred method in this study.

Chapter 5

Parameter Estimation

The system identification problem can be expressed as a cost function (Equation 5-1) that compares the likeness of the derived model (y) and the actual physical system (z). Deriving the best model based on this cost function could lead to long, tedious research on a huge amount of model candidates. Therefore, the system identification problem is modified into an optimisation problem where the goal is to find a model **M** such that it minimises the cost function. The models could be such that they have the same structure but only different by the values of the parameters, thereby reducing the system identification problem to one of parameter estimation (Klein and Morelli [2006]).

$$\begin{aligned} &J = J(z, y) \\ &z = y + v \\ &\textit{where } v \textit{ is the error between the physical system and the model} \end{aligned} \tag{5-1}$$

Parameter estimation is a process of finding the unknown parameter values in a model structure, from a set of noisy measurements. The technique for estimation depends on the noise in the measurements, if the parameters are constants or variables, and the assumption of linearity. This chapter is a discussion on the different methods of parameter estimation for time-domain systems.

Regression is a statistical technique for modelling and studying the relationship between variables. When discussing linear regression, the linearity refers to the equation for the output with respect to the parameters and not the independent variables. In other words, a linear regression model can have both linear and non-linear functions of the independent variables (Klein and Morelli [2006]).

$$y = \theta_0 + \sum_{j=1}^{n} \theta_j \xi_j \tag{5-2}$$

In Equation 5-2, the functions ξ_j are functions of the independent variables $x_1, x_2, ..., x_m$, and the model parameters $\theta_0, \theta_1, ..., \theta_m$ are constants describing the influence of each variable on the dependent variable y. However, in practice, noise must also be taken into account for the measured values of the dependent variable, as depicted in Equation 5-3, where z(i) depicts the output measurements and N is the number of data points.

$$z(i) = \theta_0 + \sum_{j=1}^{n} \theta_j \xi_j(i) + v(i) \quad i = 1,\ 2, . . . , N \tag{5-3}$$

The goal of regression methods is to find an estimate of the model parameters (θ_j). The following sections describe some common methods of regression that have also been used for modelling the systems in chapter 4.

5-1 Ordinary Least Squares

Equations 5-2 and 5-3 can be rewritten as

$$y = X\theta \tag{5-4}$$

$$z = X\theta + v \tag{5-5}$$

where
z is a vector of N measurements z(1), z(2), ..., z(N)
θ is a vector of size n+1 of the unknown model parameters $\theta_0,\ \theta_1, ...,\ \theta_n$
X is a (N x n+1) matrix of ones and regressors [1 ξ_1 ... ξ_n]
v is a vector of the measurement noise of size N v(1), v(2), ... ,v(N)
In this case, the vector v is assumed to have a zero mean and is uncorrelated, with a constant variance σ^2.

The best estimate in a least squares problem is obtained by minimising the sum of squared differences between the measured values and the model values, depicted in Equation 5-6 (Klein and Morelli [2006]).

$$J(\theta) = \frac{1}{2}(z - X\theta)^T(z - X\theta) \tag{5-6}$$

To minimise the function J, the derivative of J with respect to θ at the minimum value $\hat{\theta}$ must be equal to zero.

$$\frac{\partial J}{\partial \theta} = -X^T z + X^T X\hat{\theta} = 0 \tag{5-7}$$

Therefore,

$$X^T z = X^T X\hat{\theta} \tag{5-8}$$

$$X^T(z - X\hat{\theta}) = 0 \tag{5-9}$$

The above equations are rewritten to obtain the formula for the ordinary least squares estimator,

$$\hat{\theta}_{OLS} = (X^TX)^{-1}X^Tz \tag{5-10}$$

The covariance of the parameter estimate can also be obtained using Equation 5-11.

$$Cov(\hat{\theta}) = E[(\hat{\theta}_{OLS} - \theta)(\hat{\theta}_{OLS} - \theta)^T] = \sigma^2(X^TX)^{-1} \tag{5-11}$$

5-2 Weighted Least Squares

In the OLS method, the noise vector is assumed to have a constant variance throughout. But that is not the case for actual sensor measurements. Assuming that the variance for the sensor measurements can be obtained, the weighted least squares (WLS) method would provide a more accurate estimate of the parameters. Consider W to be a diagonal noise scaling matrix of size NxN.

$$W = diag(\sigma_1^2, \sigma_2^2, ..., \sigma_N^2) = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_N^2 \end{bmatrix} \tag{5-12}$$

The error in Equation 5-5 would become $W^{\frac{1}{2}}v$, from which the new equation for the measurement error can be obtained

$$v = W^{-\frac{1}{2}}(z - X\theta) \tag{5-13}$$

On solving the equations in a similar manner as the OLS, the formula for the weighted least squares estimate can be derived as seen in Equation 5-14.

$$\hat{\theta}_{WLS} = (X^TW^{-1}X)^{-1}X^TW^{-1}z \tag{5-14}$$

The covariance matrix for WLS is given by

$$Cov(\hat{\theta}_{WLS}) = E[(\hat{\theta}_{WLS} - \theta)(\hat{\theta}_{WLS} - \theta)^T] = (X^TW^{-1}X)^{-1} \tag{5-15}$$

5-3 General Least Squares

For more practical purposes, the assumption of homogeneous variances is also not valid. Therefore, in this case let the covariance of the errors be V, such that V is a non-diagonal matrix unlike W in the WLS method. V is non-singular and positive definite.

$$z = X\theta + v \tag{5-16}$$

$$E(v) = 0,\ Cov(v) = E(vv^T) \equiv V \tag{5-17}$$

The cost function to be minimised for the general least squares (GLS) method is

$$J_{GLS}(\theta) = \frac{1}{2}(z - X\theta)^T V^{-1}(z - X\theta) \tag{5-18}$$

The formula for the minimum parameter estimate is

$$\hat{\theta}_{GLS} = (X^T V^{-1} X)^{-1} X^T V^{-1} z \tag{5-19}$$

The covariance matrix for GLS is given by

$$Cov(\hat{\theta}_{GLS}) = E[(\hat{\theta}_{GLS} - \theta)(\hat{\theta}_{GLS} - \theta)^T] = (X^T V^{-1} X)^{-1} \tag{5-20}$$

5-4 Non-Linear Least Squares

It is not always possible to have a linear relationship between the dependent variable and the regressors. In such cases, the model is formulated as described in Klein and Morelli [2006]

$$z = h(\theta) + v \tag{5-21}$$

As a regression model Equation 5-21 can be expressed as

$$z(i) = f[x(i), \theta] + v(i) \quad i = 1,\ 2, \ldots, N \tag{5-22}$$

In Equation 5-22, f is the non-linear function of x(i) and the model parameters θ, and $x^T(i)$ refers to the row of regressors measured at the i^{th} data point.

Similar to the method used in section 5-1, the cost function to be minimised can be obtained by minimising the sum of squared differences between the model values and the measured values.

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{N}(z(i) - f[x(i), \theta])^2 \tag{5-23}$$

The minima of the cost function can be obtained by equating the derivative of the cost function to zero at the point $\theta = \hat{\theta}$

$$\left.\frac{\partial J}{\partial \theta}\right|_{\theta=\hat{\theta}} = -\sum_{i=1}^{N}(z(i) - f[x(i), \theta])\left.\frac{\partial f[x(i), \theta]}{\partial \theta}\right|_{\theta=\hat{\theta}} = 0 \tag{5-24}$$

where $\partial J/\partial\theta$ is a row vector of the partial derivative of J(θ), and $\partial f[x(i), \theta]/\partial\theta$ is the vector of sensitivity to change of the model parameters. Since Equation 5-24 is a set of non-linear algebraic equations, obtaining the value of the parameter estimate $\hat{\theta}$ cannot be done using simple algebra as done in linear regression methods. Therefore, an iterative non-linear optimisation technique like maximum likelihood estimation must be applied Klein and Morelli [2006].

Chapter 6

Synthesis of the Literature

Autonomous drone racing is becoming more agile but at the same time, it is also increasing the risk for the drone. The aim of this research is to model a high-speed racing drone in order to simulate the responses of the quadrotor with high accuracy. This will provide a basis for simulating the response without risking any damage to the drone or any property or people around the flying arena.

The literature begins with the concept of modelling a quadrotor using an understanding of the dynamics and the equations that follow and system identification method to relate the control inputs to the states and outputs of the quadrotor. In order to improve the accuracy of predicting the states using a dynamic model, various aerodynamic effects must be taken into account. However, this makes the system of equations long and complex. Therefore, for more accurate modelling of a quadrotor's response to its inputs, system identification is a preferred method.

System identification makes use of real-time experimental data that can either be used to fill in parameters in an equation (gray-box model) or establish a relationship between the inputs and outputs with no understanding of the structure of the system (black-box model).

The idea of this research is to break down the quadrotor control system into several subsystems to be identified. The first subsystem is the rate controller with the RC commands as the input and the motor response (speed of the motor) as the output. The second is the motor speeds to the states of the system (position, orientation, translational and rotational velocities) and their time derivatives (linear and rotational accelerations) using an understanding of quadrotor dynamics. Several researchers have performed system identification experiments to identify these relationships. However, the experiments are either performed on static set-ups or at slow speeds where the near-hover condition can be used to linearise the identified model.

A way to obtain the actual states in this research is using a motion tracking system. The rest of the data (control inputs, motor responses, and IMU data) can be obtained from the on-board sensors. Since these two sets of data are obtained at different frequencies, a method to align the data from different sensors must be employed. A simple solution employed by

Schubert et al. [2018] involves the calculation of a time shift and the bias once the two datasets are obtained in the same frame of reference.

Since this research aims to model a high-speed racing drone, aerodynamic effects cannot be ignored and a linear model cannot be used to describe the system. Therefore, the data to be used for system identification will be obtained from high-speed real-time free-flight experiments. At very high speeds, motors do get saturated and cannot increase their speed any further. In this case, controllers usually prioritise between the throttle commands and the rate commands. The two ways in which this can be identified is understanding the software of the controller, or analysing the real-time data obtained. In order to simulate an accurate response of the quadrotor with its controller, it is important to understand its behaviour in the case of motor saturations.

Part III

Appendix

Appendix A

Process of System Identification

The steps involved in the system identification process are discussed in this chapter. These steps form a cycle and can be run through multiple iterations until a model of required accuracy and complexity is achieved.

Selecting the Input Signals

The inputs are used to control the dynamic response of the system. Analysing the inputs and the corresponding response of the system can help identify the model structure and the model parameters. Appropriate inputs need to be selected for the test data such that all the dynamic modes are excited and the maximum values are reached (or can be estimated with high precision). But just exciting the dynamic modes might not provide the required information. Sometimes, the signals need to be excited for a longer period of time in order to identify the modes. Remple and Tischler [2006] have developed an automated frequency sweep approach that are constant or varying, depending on the dynamic response being identified. This helps separate information that can be used for system identification from the data that could be caused by a minor disturbance (Mettler [2013]).

Collection of Test Data

The main challenges involved in this aspect are identifying the parameters to be used for modelling as well as handling of the measured data obtained. The purpose of the model, and the on-board and off-board (like motion capture systems) sensors could be the factors involved in determining the output parameters to be used.

Depending on the sensors used, the measured data could be noisy. To tackle the issue of noise, using simple filters like low-pass filters could provide an adequate solution Remple and Tischler [2006]. If not, more complex methods such as Kalman filtering could be utilised. The frequency rates of the sensors must also be taken into account. Some sensors could have sample rates lower than the dynamic modes of the system, and therefore would not be able to identify the model parameters. In other cases, different sensors have different sampling rates. Therefore, if multiple sensors are used, sensor fusion might be required in order to use the measured data correctly.

Model Structure

This is an important element in system identification as the selection of an incorrect model might lead to only partially describing the dynamics of the system. Selecting the model structure is selecting the type of function that defines the output in terms of the input parameters. The structure selected is based on whether the models are parametric or non-parametric. Parametric models could either be polynomial functions like simple polynomials, or a polynomial spline function. Examples of non-polynomial models would be neural network or fuzzy logic models. For non-parametric methods, usually kernel models like support vector machines are used.

Selection of the System Identification Method

System identification methods have been grouped into parametric and non-parametric methods (Remple and Tischler [2006]). Parametric methods require a structure for the dynamic model, and the parameters are identified using the test data. Non-parametric methods do not require any information about the system dynamics and can obtain a model from impulse response or frequency responses, also obtained using the experimental values.

Parametric methods are further divided into groups as seen in Figure A-1. The figure does not represent the different types of methods used for system identification. They are divided into two methods - state estimation and parameter estimation. These are covered in more detail in chapter 5.

Validation of the Model

The model obtained from system identification must be validated with the actual test flight data. If the model is not able to predict the responses with minimal error when compared to the real quadrotor, the collected data, model structure, or the method of identification must be reconsidered.

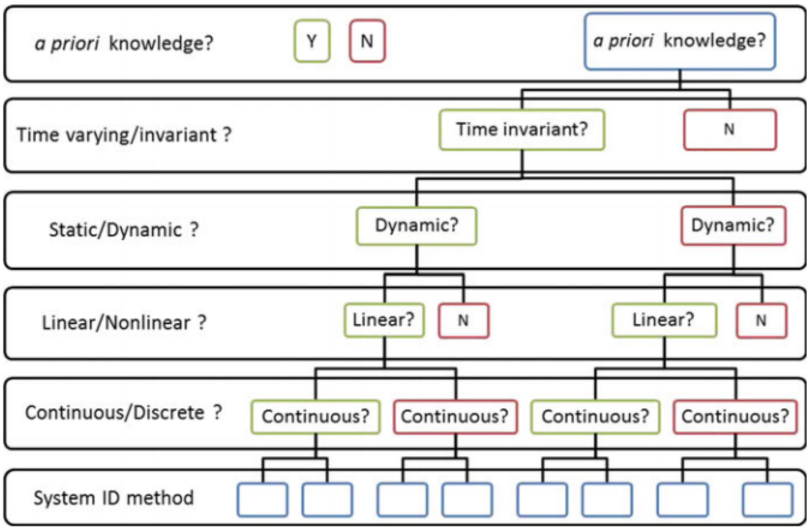


Figure A-1: Identifying the system identification method (Hoffer et al. [2014])

Bibliography

Arfita Yuana Dewi Al Al, Antonov Bachtiar, and Dwi Harinita. An angle speed and thrust relationship of the quadcopter rotor. *Indonesian Journal of Electrical Engineering and Computer Science*, 13(2):469–474, 2019.

Moses Bangura, Robert Mahony, et al. Nonlinear dynamic modeling for high performance control of a quadrotor. *Australian Robotics and Automation Association*, 2012.

Hossein Bolandi, Mohammad Rezaei, Reza Mohsenipour, Hossein Nemati, and Seed Majid Smailzadeh. Attitude control of a quadrotor with optimized pid controller. *Scientific Research Publishing*, 2013.

Samir Bouabdallah. Design and control of quadrotors with application to autonomous flying. Technical report, Epfl, 2007.

Pierre-Jean Bristeau, Philippe Martin, Erwan Salaün, and Nicolas Petit. The role of propeller aerodynamics in the model of a quadrotor uav. In *2009 European Control Conference (ECC)*, pages 683–688. IEEE, 2009.

Luis Rodolfo García Carrillo, Alejandro Enrique Dzul López, Rogelio Lozano, and Claude Pégard. *Quad rotorcraft control: vision-based hovering and navigation*. Springer Science & Business Media, 2012.

Abhijit Das, Kamesh Subbarao, and Frank Lewis. Dynamic inversion with zero-dynamics stabilisation for quadrotor control. *IET control theory & applications*, 3(3):303–314, 2009.

T Deif, A Kassem, and G El Baioumi. Modeling and attitude stabilization of indoor quad rotor. *International Review of Aerospace Engineering (IREASE)*, 7(2):43–47, 2014.

Laloui Derafa, Tarek Madani, and Abdelaziz Benallegue. Dynamic modelling and experimental identification of four rotors helicopter parameters. In *2006 IEEE International Conference on Industrial Technology*, pages 1834–1839. IEEE, 2006.

M Gonzalez-Sanchez, L Amezquita-Brooks, Eduardo Liceaga-Castro, and P del C Zambrano-Robledo. Simplifying quadrotor controllers by using simplified design models. In *52nd IEEE Conference on Decision and Control*, pages 4236–4241. IEEE, 2013.

Gregory Gremillion and James Humbert. System identification of a quadrotor micro air vehicle. In *AIAA Atmospheric Flight Mechanics Conference*, page 7644, 2010.

Nathan V Hoffer, Calvin Coopmans, Austin M Jensen, and YangQuan Chen. A survey and categorization of small low-cost unmanned aerial vehicle system identification. *Journal of Intelligent & Robotic Systems*, 74(1-2):129–145, 2014.

Gabriel Hoffmann, Haomiao Huang, Steven Waslander, and Claire Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *AIAA guidance, navigation and control conference and exhibit*, page 6461, 2007.

Haomiao Huang, Gabriel M Hoffmann, Steven L Waslander, and Claire J Tomlin. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In *2009 IEEE international conference on robotics and automation*, pages 3277–3282. IEEE, 2009.

Wayne Johnson. *Helicopter theory.* Courier Corporation, 2012.

Gaetan Kerschen, Keith Worden, Alexander F Vakakis, and Jean-Claude Golinval. Past, present and future of nonlinear system identification in structural dynamics. *Mechanical systems and signal processing*, 20(3):505–592, 2006.

Vladislav Klein and Eugene A Morelli. *Aircraft system identification: theory and practice.* American Institute of Aeronautics and Astronautics Reston, VA, 2006.

Bernard Mettler. *Identification modeling and characteristics of miniature rotorcraft.* Springer Science & Business Media, 2013.

Hyungpil Moon, Jose Martinez-Carranza, Titus Cieslewski, Matthias Faessler, Davide Falanga, Alessandro Simovic, Davide Scaramuzza, Shuo Li, Michael Ozo, Christophe De Wagter, et al. Challenges and implemented technologies used in autonomous drone racing. *Intelligent Service Robotics*, 12(2):137–148, 2019.

Eugene Morelli. System identification programs for aircraft (sidpac). In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, page 4704, 2002.

Sumaila Musa. Techniques for quadcopter modeling and design: A review. *Journal of Unmanned System Technology*, 5(3):66–75, 2018.

Paul Pounds, Robert Mahony, and Peter Corke. Modelling and control of a large quadrotor robot. *Control Engineering Practice*, 18(7):691–699, 2010.

Caitlin Powers, Daniel Mellinger, Aleksandr Kushleyev, Bruce Kothmann, and Vijay Kumar. Influence of aerodynamics and proximity effects in quadrotor flight. In *Experimental robotics*, pages 289–302. Springer, 2013.

Robert K Remple and Mark B Tischler. *Aircraft and rotorcraft system identification: engineering methods with flight-test examples.* American Institute of Aeronautics and Astronautics, 2006.

Victor Manuel Aboytes Reséndiz and Edgar A Rivas-Araiza. System identification of a quadrotor in x configuration from experimental data. *Res. Comput. Sci.*, 118:77–86, 2016.

David Schubert, Thore Goll, Nikolaus Demmel, Vladyslav Usenko, Jörg Stückler, and Daniel Cremers. The tum vi benchmark for evaluating visual-inertial odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1680–1687. IEEE, 2018.

Nikola Simidjievski, Ljupčo Todorovski, Juš Kocijan, and Sašo Džeroski. Equation discovery for nonlinear system identification. *IEEE Access*, 8:29930–29943, 2020.

Sihao Sun, Rudi Schilder, and Coen C de Visser. Identification of quadrotor aerodynamic model from high speed flight data. In *2018 AIAA Atmospheric Flight Mechanics Conference*, page 0523, 2018.

Sihao Sun, Coen C de Visser, and Qiping Chu. Quadrotor gray-box model identification from high-speed flight data. *Journal of Aircraft*, 56(2):645–661, 2019.

Pengcheng Wang, Zhihong Man, Zhenwei Cao, Jinchuan Zheng, and Yong Zhao. Dynamics modelling and linear control of quadcopter. In *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pages 498–503. IEEE, 2016.

Xiaodong Zhang, Xiaoli Li, Kang Wang, and Yanjun Lu. A survey of modelling and identification of quadrotor robot. In *Abstract and Applied Analysis*, volume 2014. Hindawi, 2014.