System identification for an autonomous drone

E. A. van Mourik M. J. van der Boon J. D. Wiersema E. P. Jongkees P. E. de Bruin

***Abstract*—With the increase in usages of drones in the research on autonomous flight, a need for precise models describing drone behavior arises. This report presents multiple dynamical models to describe the motions of the Parrot Bebop 2 commercial quadrotor in response to input. State space models of different orders are set up for the drone's input response. Translational dynamics of the drone are found from a dynamics evaluation, with resulting drag parameters to be identified. The system parameters of these models are identified from data recorded by a motion capture system during 4 separate flight experiments. The control inputs consist of random binary signals that are specifically designed to capture the drone's behaviour. Each data set results in an identified model. These models are validated using the 3 other data sets to verify their performance. The identified models with the highest average fit over the validation data sets are presented. Average NRMSE fit percentages of 83.5±1.6% for the input response are reached for a third order state space model. A second order state space model produces similar results with only marginally lower fit percentage of 82.9±0.6%. Higher order models do not seem to perform better. The application of the model will most likely dictate whether complexity or accuracy is more important and thus which order model is preferable.**

I. Introduction

Because of the unique possibilities that drones offer, multiple industries are looking into using autonomous drones for various applications, such as delivery of packages [1] and ambulance drones [2]. This increase in usages creates a demand for a way to control autonomous flight. An accurate model for the dynamics of the drone is required to plan future inputs and predict the resulting movements from those inputs. The goal of this research is to derive and identify a model that describes the dynamics of the Parrot Bebop 2 quadrotor [3] accurately. In the rest of this paper the Bebop 2 drone is referred to as quadrotor.

In the dynamics model derived for the quadrotor, multiple parameters are unknown and have to be estimated. System identification is a broad field of research and deals with the modelling of physical systems for control or optimization purposes. In this branch of research, all approaches can be roughly categorized into three categories: *white*, *grey* and *black* box system identification. These three categories are related to the level of *a priori* knowledge of the system. In white box system identification everything is known about the system and it is thus possible to derive an analytic model describing the system. The second category is grey box system identification. In this category there is some information about the system dynamics, but certain parameters are unknown [4]. In black box identification nothing is known about the system except input-output data. This data is used to fit a certain model [5].

Since its introduction, many models describing the dynamics of a quadrotor have been proposed. The most commonly used model uses the moments in the x-, y- and z-direction and the total thrust generated by the rotors as inputs [6] [7] [8]. Another approach is to use the angular speed of the four rotors as model inputs [9]. Additionally, it is possible to make a more complex nonlinear dynamics model by incorporating the aerodynamics of both the rotors and the drone itself into the model [10]. Lastly, it is an option to use the desired roll and pitch angles, the angular speed along the body z-axis and the vertical velocity as inputs of the model [11]. These are the inputs accepted by the Bebop quadrotor as described in the API [12]. A form of this model will therefore be used in this research.

This project contributes to the body of work on system identification of drones for the purpose of autonomous flight by providing multiple identified models of different orders and with varying performances for the Parrot Bebop 2 in particular as well as establishing a routine for system identification of drones in general.

The remaining part of this report is organized as follows. In section II the dynamical model of the Bebop 2 quadrotor is introduced as well as parameters that need to be identified. In section III the method to collect the data of the quadrotor's behaviour and the preprocessing of the data are discussed. Section IV presents the method used for the system identification and the results of the identification. In section V the simulation with the identified models is presented. Lastly, the methods and results are discussed in section VI, followed by section VII which concludes the report.

For the project multiple MATLAB scripts were used. These scripts, along with the data sets used, can be found on the GitHub repository [1].

II. Dynamics model

In this section the general dynamics equations for a quadrotor are derived. These equations are then particularized for the Parrot Bebop 2. The resulting set of equations is combined with an input response model for the accepted control input commands of the Bebop 2 quadrotor to create a full state space model describing its dynamics. The full state space model can be found in appendix B.

A. General rigid body dynamics

As shown in Fig. 1, two Cartesian frames are used. The world fixed frame is denoted as $\mathcal{A}$ and the body fixed frame

[1] https://github.com/mjvanderboon/bebop-identification

denoted as $\mathcal{B}$. The body fixed frame is obtained by rotating the inertial world frame by the Z-X-Y Euler angles. These angles, pitch, roll and yaw, are denoted by θ, ϕ, ψ respectively.

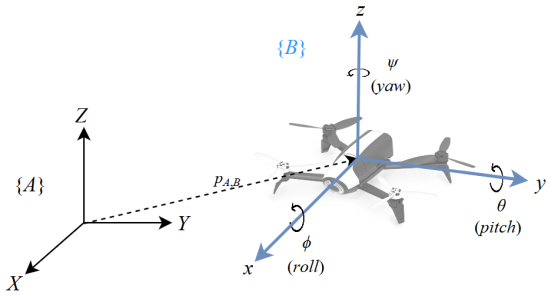

Fig. 1: Definition of frames

The combined rotation matrix from the inertial frame $\mathcal{A}$ to the body-fixed frame $\mathcal{B}$ is as follows:

$$^{\mathcal{A}}R_{\mathcal{B}} = R_Z(\psi)R_X(\phi)R_Y(\theta) \tag{1}$$

where $R_Z(\psi), R_X(\phi), R_Y(\theta) \in \mathbb{R}^3$ are rotation matrices.

The vectors $\mathbf{p}^{\mathcal{A}} = [x\ y\ z]^T$ and $\mathbf{v}^{\mathcal{A}} = [v_x\ v_y\ v_z]^T$ describe the position and velocity in the world frame.

The general translational equation of motion is given by Eq. (2) in the $\mathcal{A}$ frame.

$$m\dot{\mathbf{v}}^{\mathcal{A}} = -m\mathbf{g}^{\mathcal{A}} +^{\mathcal{A}} R_{\mathcal{B}}\mathbf{T}^{\mathcal{B}} \tag{2}$$

Where m is the quadrotor's mass, $\mathbf{g}^{\mathcal{A}} = [0 \quad 0 \quad g]^T$ is a vector for the earth's gravity and $\mathbf{T}^{\mathcal{B}} = [0 \quad 0 \quad T]^T$ is the thrust vector of the quadrotor in the $\mathcal{B}$ frame located at the center of mass.

Aerodynamic drag is incorporated in Eq. (3). To simplify the model, the relation between the drag force and the velocity is approximated to be linear. This is warranted, because the velocities of the quadrotor are relatively low, in the range of 0 to 0.5 m/s.

$$\mathbf{F}^{\mathcal{A}}_{drag} =^{\mathcal{A}} R_{\mathcal{B}} D\mathbf{v}^{\mathcal{B}} \tag{3}$$

This results in the following equation:

$$m\dot{\mathbf{v}}^{\mathcal{A}} = -m\mathbf{g}^{\mathcal{A}} +^{\mathcal{A}} R_{\mathcal{B}}\mathbf{T}^{\mathcal{B}} -^{\mathcal{A}} R_{\mathcal{B}} D\mathbf{v}^{\mathcal{B}}. \tag{4}$$

Matrix D is a diagonal matrix containing the body aerodynamic drag coefficients k_{D_x} and k_{D_y}. The drag coefficient in the z-direction is not taken into account. This is further elaborated in appendix A.

Since the angles ϕ and θ are assumed to be fairly small, the assumption in Eq. (5) is made to reduce computational load with a presumed small effect. This simplification says that the drag force components in the world frame are not influenced by pitch and roll angles of the quadrotor.

$$\mathbf{F}^{\mathcal{A}}_{drag} =^{\mathcal{A}} R_{\mathcal{B}} D\mathbf{v}^{\mathcal{B}} \approx R_Z D\mathbf{v}^{\mathcal{B}} \approx R_Z D {R_Z}^{-1}\mathbf{v}^{\mathcal{A}} \tag{5}$$

This simplifies Eq. (4) to:

$$m\dot{\mathbf{v}}^{\mathcal{A}} = -m\mathbf{g}^{\mathcal{A}} +^{\mathcal{A}} R_{\mathcal{B}}\mathbf{T}^{\mathcal{B}} - R_Z D {R_Z}^{-1}\mathbf{v}^{\mathcal{A}} \tag{6}$$

B. Quadrotor dynamics model

The inputs that can be given to the quadrotor through the Parrot software development kit (SDK) are contained in the vector $\mathbf{u}$.

$$\mathbf{u} = [\phi_c \quad \theta_c \quad \dot{\psi}_c \quad v_{zc}]^T \in \mathbb{R}^4 \tag{7}$$

where ϕ_c, θ_c are the commanded roll and pitch angles, v_{zc} is the commanded vertical velocity in the $\mathcal{A}$ frame and $\dot{\psi}_c$ is the commanded yaw rate in the $\mathcal{B}$ frame.

The quadrotor's state in the world frame is denoted by $\mathbf{y}^{\mathcal{A}}$. Here x, y and z are the positions of the quadrotor, v_x, v_y and v_z are the velocities, ϕ, θ and ψ the Euler angles with respect to the world frame.

$$\mathbf{y}^{\mathcal{A}} = [x\ y\ z\ v_x\ v_y\ v_z\ \phi\ \theta\ \psi]^T \in \mathbb{R}^9 \tag{8}$$

1) Translational dynamics: The vertical acceleration is described by the commanded vertical velocity with an input-response model as shown in Eq. (9).

$$\dot{v}_z = h_{v_z}(v_{zc}, v_z) \tag{9}$$

The thrust can be expressed as a function of the vertical acceleration and the pitch and roll angels by evaluating Eq. (6) in the Z-direction, which yields:

$$T = \frac{m(\dot{v}_z + g)}{\cos(\phi)\ \cos(\theta)} \tag{10}$$

With the vertical acceleration being described by one of the models in section II-C, Eq. (10) can be substituted in Eq. (6) to obtain the translational dynamics of the quadrotor:

$$\begin{cases} \dot{v}_x = k^*_{D_y}\sin(\psi)(v_y\cos(\psi) - v_x\sin(\psi)) \\ \qquad -k^*_{D_x}\cos(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) \\ +(\cos(\psi)\dfrac{\tan(\theta)}{cos(\phi)} + \tan(\varphi)\sin(\psi))(\dot{v}_z + g) \\ \dot{v}_y = -k^*_{D_x}\sin(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) \\ \qquad -k^*_{D_y}\cos(\psi)(v_y\cos(\psi) - v_x\sin(\psi)) \\ +(\sin(\psi)\dfrac{\tan(\theta)}{cos(\phi)} - \cos(\psi)\tan(\varphi))(\dot{v}_z + g) \end{cases} \tag{11}$$

With $k^*_{D_{...}} = k_{D_{...}}/m$. These equations are also fully derived in appendix B.

2) Rotational dynamics: All rotations are determined by the control inputs in $\mathbf{u}$. Therefore the rotational dynamics of the quadrotor are described by the following input-response models. Section II-C deals with the forms assumed for these models.

$$\dot{\phi} = h_\phi(\phi_c, \phi), \tag{12}$$

$$\dot{\theta} = h_\theta(\theta_c, \theta), \tag{13}$$

$$\ddot{\psi} = h_\psi(\dot{\psi}_c, \dot{\psi}). \tag{14}$$

Eq. (9) and Eq. (11) to Eq. (14) constitute the state space dynamical model of the quadrotor. The full state space models can also be found in appendix B.

C. Input response models

To identify the input response models in Eq. (12), (13), and (14), different models are assumed with varying parameters to be identified. In this section a first order, a first order plus time delay, and a general higher order model with and without time delay are proposed.

1) First order model: The first order input response model is described by Eq. (15):

$$\begin{cases} \dot{\phi} = \frac{1}{\tau_\phi}(k_\phi \phi_c - \phi) \\ \dot{\theta} = \frac{1}{\tau_\theta}(k_\theta \theta_c - \theta) \\ \ddot{\psi} = \frac{1}{\tau_{\dot{\psi}}}(k_{\dot{\psi}} \dot{\psi}_c - \dot{\psi}) \\ \dot{v}_z = \frac{1}{\tau_{v_z}}(k_{v_z} v_{zc} - v_z) \end{cases} \tag{15}$$

Here $k_\phi, k_\theta, k_{\dot{\psi}}$ and k_{v_z} are the steady state gains and $\tau_\phi, \tau_\theta, \tau_{\dot{\psi}}$ and τ_{v_z} are the time constants. These parameters are to be identified for this model.

2) First-order-plus-time-delay model: Since the quadrotor is a physical system to which signals are sent via a computer, multiple delays may arise between the generating of the commands, and the quadrotor executing them. Therefore, the first order model can be extended by adding a time delay (FOPTD). By incorporating the time delay, the first order model changes to:

$$\begin{cases} \dot{\phi} = \frac{1}{\tau_\phi}(k_\phi \phi_c(t - \beta_\phi) - \phi) \\ \dot{\theta} = \frac{1}{\tau_\theta}(k_\theta \theta_c(t - \beta_\theta) - \theta) \\ \ddot{\psi} = \frac{1}{\tau_{\dot{\psi}}}(k_{\dot{\psi}} \dot{\psi}_c(t - \beta_{\dot{\psi}}) - \dot{\psi}) \\ \dot{v}_z = \frac{1}{\tau_{v_z}}(k_{v_z} v_{zc}(t - \beta_{v_z}) - v_z) \end{cases} \tag{16}$$

Here β is the time delay and it is assumed that $\beta_\phi,\ \beta_\theta,\ \beta_{\dot{\psi}}$ and β_{v_z} can be different for every input.

3) General higher order model: To get a more expressive description of the system a higher order model can be used. A higher order state space model for the input response results in the following model:

$$\begin{cases} \dot{\mathbf{x}}_\phi = A_\phi \mathbf{x}_\phi + B_\phi \phi_c \\ \phi = C_\phi \mathbf{x}_\phi \\ \dot{\mathbf{x}}_\theta = A_\theta \mathbf{x}_\theta + B_\theta \theta_c \\ \theta = C_\theta \mathbf{x}_\theta \\ \dot{\mathbf{x}}_{\dot{\psi}} = A_{\dot{\psi}} \mathbf{x}_{\dot{\psi}} + B_{\dot{\psi}} \dot{\psi}_c \\ \dot{\psi} = C_{\dot{\psi}} \mathbf{x}_{\dot{\psi}} \\ \dot{\mathbf{x}}_{v_z} = A_{v_z} \mathbf{x}_{v_z} + B_{v_z} v_{zc} \\ v_z = C_{v_z} \mathbf{x}_{v_z} \end{cases} \tag{17}$$

The state variables $\mathbf{x}_\phi,\ \mathbf{x}_\theta,\ \mathbf{x}_{v_z}$ and $\mathbf{x}_{\dot{\psi}}$ are non-physical variables related to the real measured angles and vertical velocity through the control matrices C. For a second order model this results in a [2x2] A matrix, a [2x1] B matrix and a [1x2] C matrix. The unknown parameters in this model are the entries in the A, B, and C matrices.
In general this results in $4(n^2 + 2n)$ parameters to be estimated, where n is the model order, for a model where the order for all inputs is chosen to be the same. A second order model has 32 parameters, a third order model 60, etcetera.

4) General order plus time delay model: Just like the FOPTD model, a time delay can also be added to the higher order models. This results in the equation given in Eq. (18).

$$\begin{cases} \dot{\mathbf{x}}_\phi(t) = A_\phi \mathbf{x}_\phi(t) + B_\phi \phi_c(t - \beta_\phi) \\ \phi(t) = C_\phi \mathbf{x}_\phi(t) \\ \dot{\mathbf{x}}_\theta(t) = A_\theta \mathbf{x}_\theta(t) + B_\theta \theta_c(t - \beta_\theta) \\ \theta(t) = C_\theta \mathbf{x}_\theta(t) \\ \dot{\mathbf{x}}_{\dot{\psi}}(t) = A_{\dot{\psi}} \mathbf{x}_{\dot{\psi}}(t) + B_{\dot{\psi}} \dot{\psi}_c(t - \beta_{\dot{\psi}}) \\ \dot{\psi}(t) = C_{\dot{\psi}} \mathbf{x}_{\dot{\psi}}(t) \\ \dot{\mathbf{x}}_{v_z}(t) = A_{v_z} \mathbf{x}_{v_z}(t) + B_{v_z} v_{zc}(t - \beta_{v_z}) \\ v_z(t) = C_{v_z} \mathbf{x}_{v_z}(t) \end{cases} \tag{18}$$

III. Experimental overview

In this section the procedure for data collection and processing is described. First the experimental setup is elaborated on, followed by the control inputs and data processing.

A. Experimental setup

Fig. 2 provides an overview of the setup used during the experiments. A laptop is used to send and receive information to conduct and control the experiments. The type of drone used for the experiments is the Parrot Bebop 2. During all experiments the same drone is used in combination with different batteries. The drone is connected via its own WiFi network to the control laptop. To track the position and attitude of the drone a motion capture system, the Mocap Optitrack [13], is used. Control input commands are sent to the drone via WiFi using the Robot Operating System (ROS) [14] and the ROS package bebop_autonomy [15]. The full states of the drone including the velocities and accelerations are estimated using an unscented Kalman filter. This filter estimates these states using the position data measured by the Mocap system. This is also shown in Fig. 2.

The loop running on the control laptop runs at 20 Hz. This means that the input commands are sent to the quadrotor at 20 Hz. The outputs from the estimator and the Mocap are logged at 20 Hz as well. A frequency of 20 Hz was chosen, because the inner control loop of the quadrotor runs at 20 Hz [12]. A higher frequency results in a longer time delay as elaborated in appendix D-A.

B. Control inputs

A random binary signal (RBS) was selected as the best input to estimate a model valid for a wide range of applications based on a few observations. An RBS signal is a blockwave signal that is either +1 or -1 (binary) and has a randomly varying frequency for the blockwave. First of all, using an RBS makes sure that all relevant frequencies for the system are excited. The bandwidth of each input signal was determined separately for the pitch, roll and vertical velocity inputs as described in appendix D. Exciting a wide range of frequencies

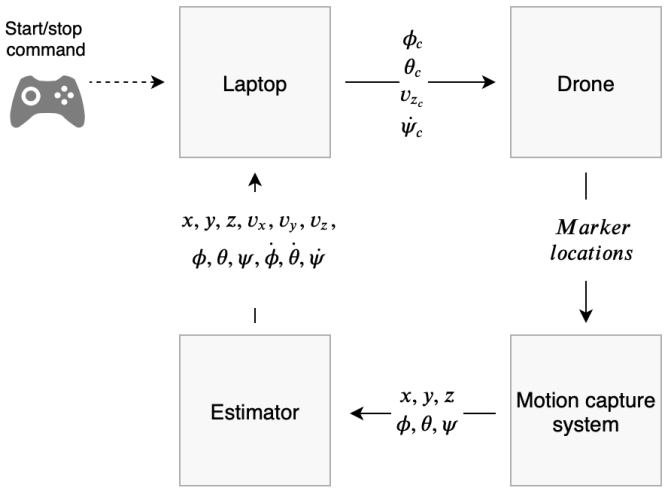

Fig. 2: Setup data flow for experiments

minimizes unwanted overfitting to the particular data set that is used for estimation.

Secondly, it can be argued that a binary signal closely represents a signal that would be used in autonomous flight. A random Gaussian noise could be suitable for this as well. However, generating a binary signal ensures that the input signal is of sufficient amplitude for the measurement of the state to be significantly different than the amplitude of any noise in the system.

The generated RBS was limited to a frequency band. The upper cutoff frequency was calculated from the bandwidth of the excited input. The lower cutoff was constrained by the dimensions of the laboratory. The input signal was sent to the quadrotor at a frequency of 20 Hz. For more information on the design of the frequency properties of the input signal see appendix D.

During the experiments all three inputs pitch, roll and vertical velocity, are excited with an RBS signal simultaneously, but the RBS signal for each input is different. The randomness of the signals causes different combinations of inputs to be excited at the same time during the experiment. This is done to represent the kind of inputs that can be expected during autonomous flight. Also, there exists no dependency between inputs, meaning there is no difference between simultaneous inputs and single input. This is shown in appendix E.

C. Data preprocessing

Before the recorded data can be used for the parameter estimation, the data needs to be preprocessed.

Firstly, the takeoff and landing of the quadrotor are cut from the data.

Secondly, the data is filtered to remove the high frequency measurement noise from the Mocap system. It is approximated that the system bandwidth is 1.35 ± 0.28 Hz as shown in appendix D-B. Applying a filter using a cutoff frequency sufficiently higher than this approximation ensures that system identification is not influenced by the filtering of the data, because the system response does not occur in those higher frequencies. Therefore, a low-pass filter with a cutoff frequency of 2 Hz was chosen.

Lastly, the data is 'deyawed'. In the first phase of the system identification the yaw rate input is set to zero. However, the quadrotor does yaw marginally during experiments. This influences the recorded data, because the position and the velocities are measured in the world frame by the Mocap system. This behaviour is identified to be random and is therefore not modeled. To correct for this the data is rotated back based on the recorded yaw. This is further discussed in section VI-A.

IV. System identification

The system identification consists of two parts, the identification of the drag coefficients and the identification of the input response model. First the drag coefficient estimation method is discussed followed by the results for the drag coefficients. Next the input response model identification method and results are discussed.

A. Drag coefficients

The drag coefficients are regarded as physical properties of the quadrotor and are therefore identified differently than the input response model parameters. For the input data over which the coefficients are estimated, step inputs for the roll and pitch are used. The parameters are estimated over the steady state part of the quadrotor's step response. Here the drag coefficients are the only parameters that influence the model. Furthermore, the orientation of the quadrotor is constant so the frontal area in the direction of travel is constant as well. To estimate the parameters grey box model identification is used. This is further discussed in section IV-B1

Multiple data sets are used for both roll and pitch to estimate the drag coefficients in x- and y-direction. Because the drag coefficients are physical properties of the quadrotor, the results are regarded as measurements and not estimations of behaviour. The final $k^*_{D_x}$ and $k^*_{D_y}$ are thus the average of the identified coefficients of each data set.

The estimated values and standard deviations for the drag coefficients are shown in table I.

	Drag $(1/s)$
$k^*_{D_x}$	0.25 ± 0.08
$k^*_{D_y}$	0.33 ± 0.05

TABLE I: Estimated drag coefficients

B. Input response model identification method

The input response model identification consists of multiple steps. A model is estimated and validated based on the preprocessed data. The resulting validation is used to select the best models.

1) Parameter estimation: To identify a model the parameters of the model need to be estimated.

To estimate the time delay, the MATLAB function `deleyest` is used. This function estimates the number of time steps it takes for the output data to respond to the input data. To estimate the parameters a non-linear grey-box estimation is used. The MATLAB function `nlgreyest`, which is part of the System Identification Toolbox, is used for the estimation. Using the input and output data the unknown parameters are then estimated. The Levenberg-Marquardt method described in appendix G is used to optimize the least squares fit.

Since it is not known whether this method converges to a local or a global minimum, multiple initial estimates for the parameters have to be generated. These initial values are chosen according to a Central Composite Design [16]. This generates a number of initial points for every parameter that needs to be estimated within a certain interval. The Central Composite Design is often used in the design of experiments [17]. For the higher order models (above 1^{st}), however, Central Composite Design is not applied. The higher order models have a large number of parameters that need to be estimated. This would result in too many initial sample points that need to be run for the processing power available. Instead, random points are generated along the interval, which reduces the amount of points to be run. More on Central Composite Design and the design of experiments can be found in appendix H.

2) Model validation: In order to determine how well a model actually estimates the quadrotor's real behaviour, the model is validated by comparing its response to measured data.

To be able to validate the model, the model is simulated for a given input set using the `sim` function in MATLAB. This function solves the differential equations of the identified model using the Runge-Kutta(4,5) numerical integration method. Different integration methods were tested. It is concluded that there is no significant difference observed between the possible numerical integration methods. More on this can be found in appendix I. For the validation, the output of this simulation is compared to measured data from the same input set.

For quantitatively scoring the validation of a particular model the `goodnessOfFit` function in MATLAB is used to compute the normalized-root-mean-square error (NRMSE) with regards to the measured data. This error is used to measure the accuracy of the model compared to the measured data. For each input/output pair (ϕ_c/ϕ, θ_c/θ & v_{zc}/v_z) a fit percentage is returned which is equal to $1 - NRMSE$. The score of a particular identified model for a particular validation data set is the average of these three percentages.

For the validation of a model multiple data sets are used. How the identified models and data sets were used to pick a final model is further elaborated in the following section.

3) Parameter set selection: For each data set the parameters of a model can be estimated as described in section IV-B1. Multiple experiments and thus multiple data sets result therefore in multiple identified models of the same type. The parameter estimation is performed on the data set of each experiment separately. In order to pick the best identified model, every model is validated with all the original data sets except the data set used for the estimation of that model using the method described in section IV-B2. By taking the average of these validation scores an overall score for each identified model is found as well as the standard deviation of the score. This score approximates how accurate an identified model is for an arbitrary input set. The overall scores for each model are compared and the model that has the highest score is the best model, since it has the highest average fit over all data. This method is depicted in Fig. 3.

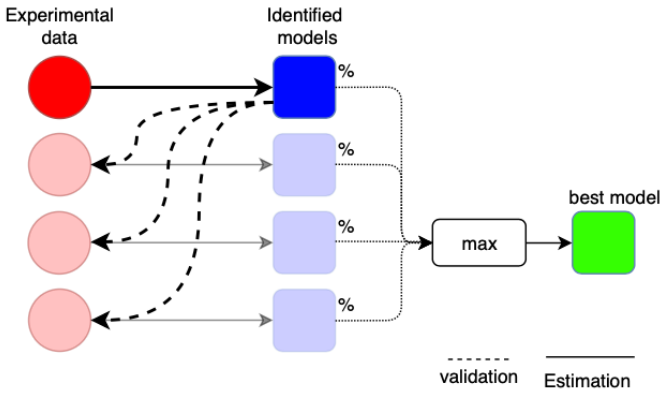

Fig. 3: Method used to select identified model with best fit.

The identified models are validated using all original data sets because this will minimize the chance of overfitting on one particular data set. The model that turns out the best with this method will be the best in estimating the response for different inputs.

4) Model comparison: After the best model for each type is identified using the method described in IV-B3 a similar method can be applied to find the best overall model out of all different types. The overall fit scores for the best models of each type are compared. However, in this case the best fit score does not necessarily mean that this is the best model, because the complexity of the model needs to be taken into account as well. The complexity of the model is important when the application is taken into account. One application could be the use in a model predictive controller (MPC) which has to be able to evaluate the model as quickly as possible. An overly complex model will need too much computational time and will reduce the speed at which the controller can send commands to the quadrotor which is an important limitation.

C. Input response model identification results

The resulting overall fit scores of the best model of each type are given in table II. The fit score used for the comparison is discussed in IV-B3 and is $(1 - NRMSE) \cdot 100\%$. These fit percentages are calculated from simulating the response of the best performing model of every model type over all data sets except the set the model was identified over, as described in section IV-B3.

The higher order models with time delay did not give significant improvements, but the results for these models can be found in appendix C.

Model	fit (%)
1^{st}	73.4±2.6
1^{st} + TD	76.7± 2.8
2^{nd}	82.9± 0.6
3^{rd}	83.5± 1.6
4^{th}	82.3± 4.5

TABLE II: Average fit of best identified model

The identified models for the first order plus time delay, second order and third order model are given below.

First order plus time delay model

$$\dot{\phi}(t) = \frac{1}{0.1598}(0.9655 \cdot \phi_c(t-0.05) - \phi(t))$$
$$\dot{\theta}(t) = \frac{1}{0.1621}(0.9795 \cdot \theta_c(t-0.05) - \theta(t))$$
$$\dot{v}_z(t) = \frac{1}{0.2599}(1.1635 \cdot v_z(t-0.05) - v_z(t))$$

Second order model

$$\dot{\mathbf{x}}_\phi = \begin{bmatrix} -3.6268 & 4.5680 \\ -4.2580 & -3.8136 \end{bmatrix} \mathbf{x}_\phi + \begin{bmatrix} 3.3057 \\ -1.5443 \end{bmatrix} \phi_c$$

$$\dot{\mathbf{x}}_\theta = \begin{bmatrix} -6.5559 & 5.4182 \\ -4.8719 & -5.1334 \end{bmatrix} \mathbf{x}_\theta + \begin{bmatrix} -0.2825 \\ -3.9654 \end{bmatrix} \theta_c$$

$$\dot{\mathbf{x}}_{v_z} = \begin{bmatrix} -4.5702 & 4.4186 \\ -4.9402 & -4.7852 \end{bmatrix} \mathbf{x}_{v_z} + \begin{bmatrix} 3.3742 \\ 0.4442 \end{bmatrix} v_{z_c}$$

$$\phi = \begin{bmatrix} -0.1337 & -1.4979 \end{bmatrix} \mathbf{x}_\phi$$

$$\theta = \begin{bmatrix} -2.2622 & -0.1050 \end{bmatrix} \mathbf{x}_\theta$$

$$v_z = \begin{bmatrix} 0.1768 & -2.5312 \end{bmatrix} \mathbf{x}_{v_z}$$

Third order model

$$\dot{\mathbf{x}}_\phi = \begin{bmatrix} -3.0696 & -3.5650 & -2.4296 \\ 0.0520 & -5.4143 & -5.9011 \\ 3.7089 & 1.0430 & -1.4279 \end{bmatrix} \mathbf{x}_\phi + \begin{bmatrix} -2.1462 \\ 1.6727 \\ 3.5351 \end{bmatrix} \phi_c$$

$$\dot{\mathbf{x}}_\theta = \begin{bmatrix} -6.1824 & -1.8848 & -5.5527 \\ -2.7956 & -1.6305 & 2.0099 \\ 2.9196 & 2.5074 & -6.8438 \end{bmatrix} \mathbf{x}_\theta + \begin{bmatrix} 3.2526 \\ 1.1928 \\ -0.7547 \end{bmatrix} \theta_c$$

$$\dot{\mathbf{x}}_{v_z} = \begin{bmatrix} -4.4829 & 4.6997 & 8.1645 \\ -3.2938 & -5.6196 & 2.7175 \\ -7.0201 & -1.7885 & -3.5407 \end{bmatrix} \mathbf{x}_{v_z} + \begin{bmatrix} 1.4509 \\ -3.2656 \\ 3.3287 \end{bmatrix} v_{z_c}$$

$$\phi = \begin{bmatrix} -0.5915 & -0.8319 & 0.4358 \end{bmatrix} \mathbf{x}_\phi$$

$$\theta = \begin{bmatrix} 1.5283 & -0.8466 & 3.8820 \end{bmatrix} \mathbf{x}_\theta$$

$$v_z = \begin{bmatrix} 0.5327 & -2.8051 & -2.4470 \end{bmatrix} \mathbf{x}_{v_z}$$

The input responses of these model are shown in Fig. 4, 5 and 6. For illustration purposes the data set that is used is the one the model fits best to. Therefore, the average fit percentage may be higher than the average fitness presented in table II.

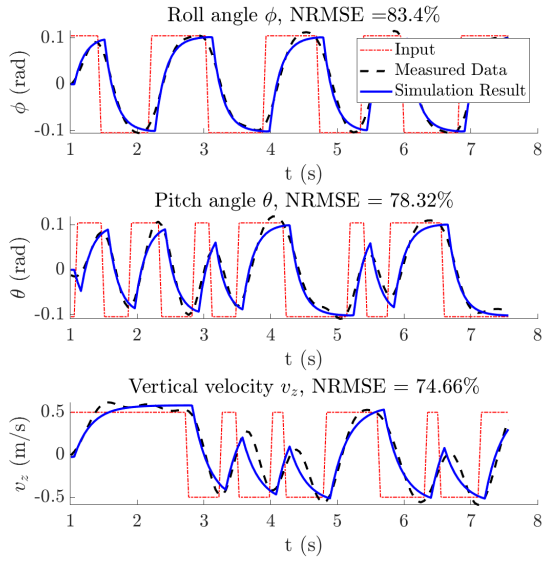

Fig. 4: Input response for 1^{st} order plus time delay model

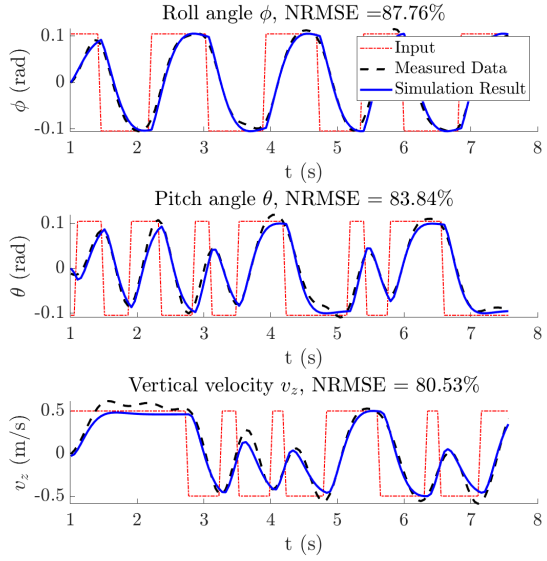

Fig. 5: Input response for 2^{nd} order model

V. Simulation

The quadrotor's movements can be simulated using an identified model. For this the complete model of the dynamics

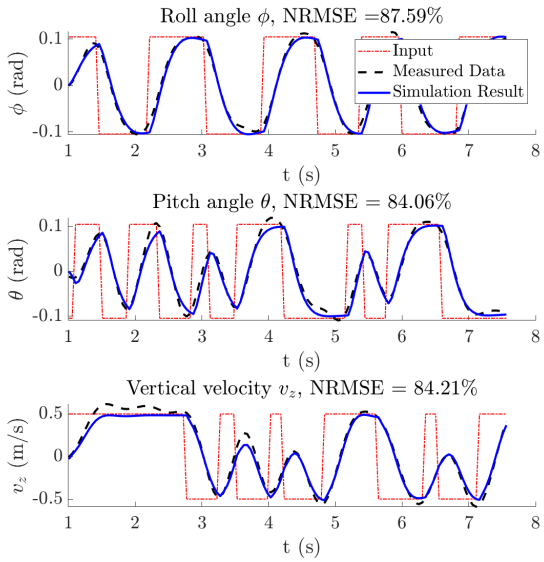

Fig. 6: Input response for 3^{rd} order model

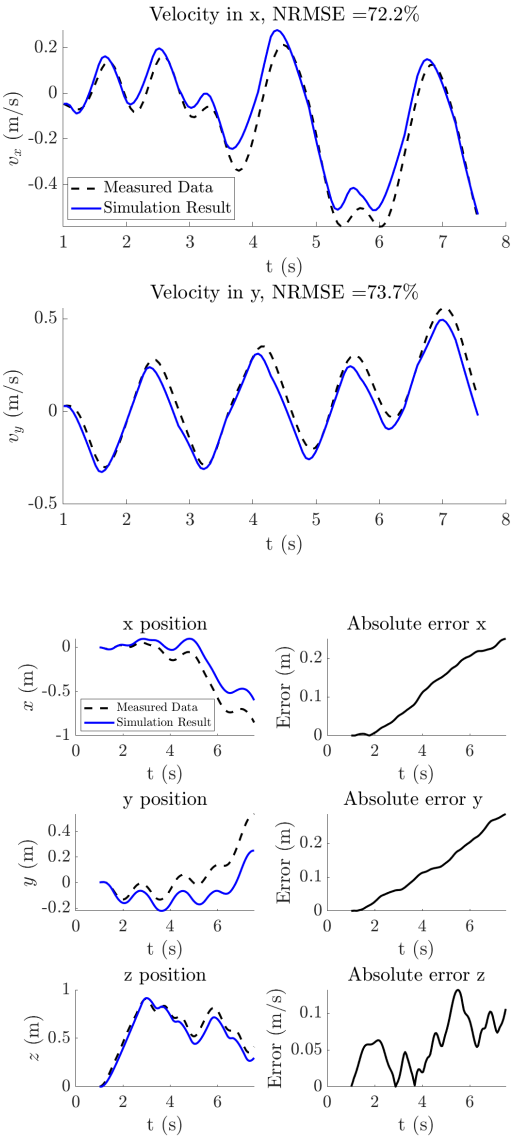

Fig. 7: Simulation results of velocity and position dynamics for 1^{st} order plus time delay model over best fitting data set

is used which is a combination of the quadrotor dynamics model and the input response model. This is further elaborated in appendix B. For the simulation the MATLAB function `sim` is used. This function simulates the model with the given initial states and input with an infinite prediction horizon. The prediction horizon states the moment that measurement data is used to correct the simulation. Because there are no modeled disturbances, the models are of Output-Error structure and thus there is no difference between simulation and prediction [18]. Therefore only simulation is used.

For an accurate simulation the initial states need to be known. For the first order models all the states are physical properties that can be measured, so the initial states are known. For the initial states of the higher order models this is not the case, so they need to be estimated. This state estimation is done using an Extended Kalman Filter. This Kalman Filter is further discussed in appendix K. Once the state estimation has converged and the initial states are known, the simulation is started. This is the reason the simulation of the higher order models does not start at $t = 0$ s.

The simulation results for the best identified first order plus time delay, second order and third order model are given in Fig. 7, 8 and 9. The simulation results of the other identified models can be found in appendix L.

VI. Discussion

In this section, possible general sources of error are discussed to get an understanding of the uncertainties in the results.

A. Unwanted yaw angles

During the conducted experiments the commanded yaw rate is kept at zero. This choice is made because the quadrotor can reach every position in the $\mathbb{R}^3$ space with the roll, pitch and vertical velocity as inputs. Furthermore, the identification of the input response models of the other parameters do not require the yaw to be considered. However, during flying the quadrotor shows a yaw drift. This yaw drift has an influence on the system identification because the Mocap system measures the position of the quadrotor in the world frame. A consequence of this is that a pure pitch will result in both a change of X- and Y-direction instead of merely a change in X-direction. To correct for this the data is deyawed, as mentioned in section III-C. The direction of the yaw drift seems to be random, as shown in Fig. 10. Also, it was observed that the battery might have an influence on the yaw drift. The yaw drift and deyawing of the data are further elaborated in

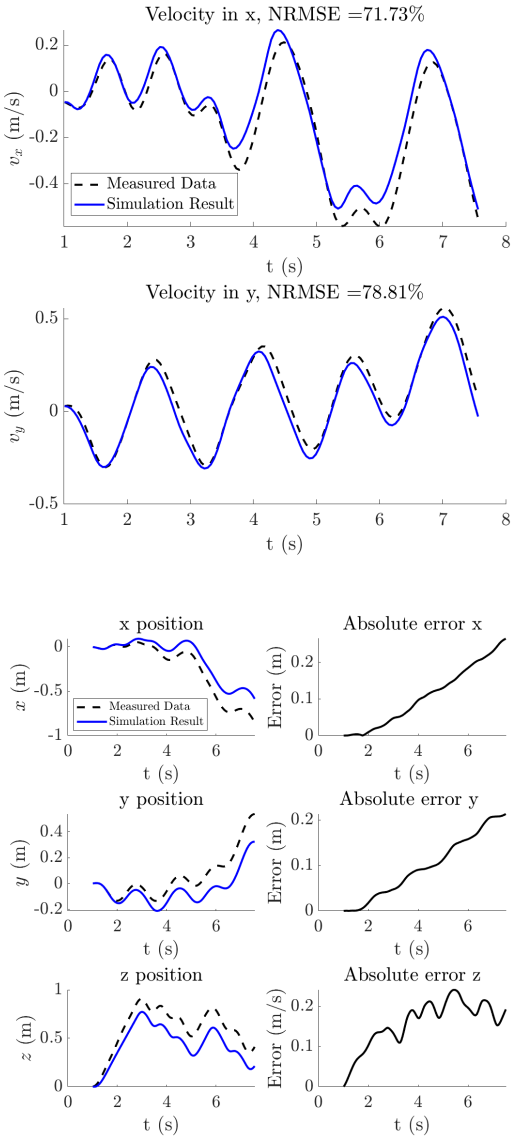

Fig. 8: Simulation results of velocity and position dynamics for 2^{nd} order model over best fitting data set

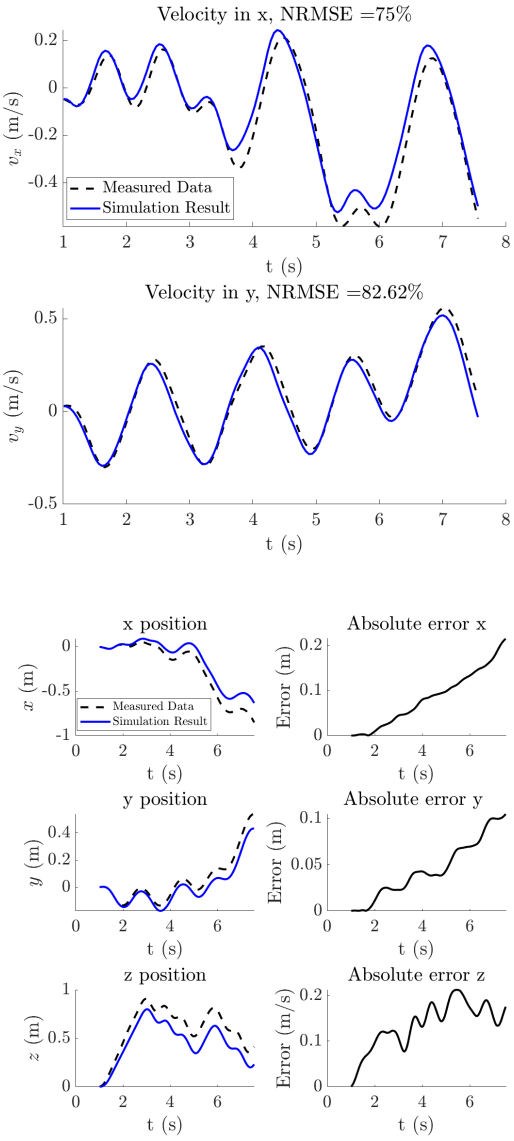

Fig. 9: Simulation results of velocity and position dynamics for 3^{rd} order model over best fitting data set

appendix F.

Despite not experimenting with the yaw, a yaw input-response model is added to the complete state space model of the quadrotor dynamics. In further research this model could be identified using the same methods as discussed in this report and added to the models that have already been identified. This yaw identification is briefly discussed in appendix J.

B. Low-pass filter

To filter out unwanted high frequency noise, a low pass filter is used. This could filter out real dynamics which could in terms influence the model accuracy. To prevent this, the cutoff frequency is chosen well above the system bandwidth as shown in appendix D-B. It could, however, be that this filtering does still influence some of the system dynamics.

C. Drag coefficients

The results obtained for $k^*_{D_x}$ and $k^*_{D_y}$ presented in section IV-A are different from the results obtained in [19]. Since the mass of the drone is about 0.5 kg, the real k_{D_x} and k_{D_x} lie around 0.125 and 0.1515 respectively. This is a factor 2 lower then the values used in [19], which are around 0.28. This could be due to the fact that these values are obtained through system identification, and in [19] the values are obtained from literature. A possibility is thus that the inputs used to determine the drag coefficients are not sufficient. It could also be that the turning of the rotors and the thrust generated results in the

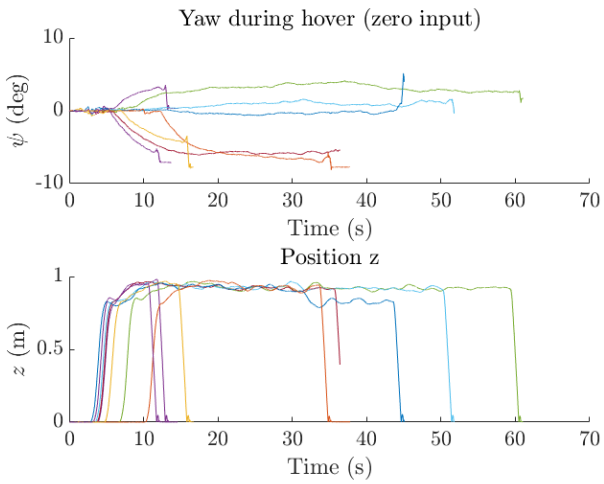

Fig. 10: Drift in yaw occurring during long experiments without input

difference in the drag coefficient, since this is not taken into account in [19].

Since the standard deviation of the drag coefficients is quite high, around 30%, it could also be that the drag coefficients do not have a very big impact on the equations for $\dot{v}_x$ and $\dot{v}_y$. This could result in this high standard deviation, because the drag coefficient will be varied a lot in the optimization process.

The model also assumes a linear relation between the drag force and the velocity. In reality, this is a quadratic relationship. This could also be a reason for the large deviation. In future research a quadratic relationship between the drag force and the velocity could be used to improve the estimates for the drag coefficients.

D. Model identification

In table II the average fit scores are given for multiple models. It is shown that the fit percentages for the 2^{nd}, 3^{rd} and 4^{th} order seem to have converged since the difference between the models is less then 5%. There is no significant difference between these models. However, the difference between the 1^{st}, $1^{st}+TD$ and 2^{nd} models is significant.

It can also be seen that these percentages are lower then the percentages obtained by N. Podhar in [19]. This deviation could be due to various reasons.

1) Initial samples: To be able to know with certainty that the optimization algorithm has converged to a global minimum (within a set range), multiple initial samples for the estimation parameters have to be tried.However, there is no absolute certainty that the optimization has converged to a global minimum. If more time is available in future research the Central Composite Design method could be applied to ensure that the found models are indeed a global minimum.

2) Time delay resolution: In the code used, sending commands and measuring data happens in the same loop. This results in measurement data with a frequency of 20 Hz. The time delay estimation method discussed in IV-B1 can only estimate the delay as $n * dT$ where n is an integer and dT is the sample time. The resolution for the time delay estimation is therefore the same as the input frequency. By decoupling the measurement from the input, this resolution can be increased and a better estimation of the time delay can be achieved.

3) System inputs: The inputs that are given to the quadrotor as described in section III-B might not be able to excite the entire system dynamics bandwidth. It is shown, however, that the inputs used in this report are more complex and excite more frequencies then the inputs used in [19]. This is likely one of the reasons the fit percentages presented in table II differ so much from the fit percentages presented in [19].

4) Estimation data sets: To generate different models of the same type, multiple data sets are used. These data sets all have an RBS on every input except the yaw rate. Since these data sets are random, more accurate results could be obtained by lengthening the inputs. This would increase the accuracy of the individual identified models since the parameters would converge to the true value. It is, however, not possible to give an infinitely long input due to the limited space in the lab.

It could also be that a model generated on a specific estimation data set is over fitted to this type of input, and will perform less on different inputs. This is minimized by using an RBS signal as input, but can of course not completely be excluded.

5) Thrust saturation: Since the amount of thrust that can be produced per motor is limited, it can occur that a motor cannot produce enough thrust to carry out one or more commands at the same time. If the vertical velocity and the pitch for example are excited at the same time, it could happen that the thrust required from the rear motors exceeds the maximum thrust. This could cause non-linearities in the measured data. In appendix M it is shown, however, that this does not occur during the experiments. Nonetheless, thrust saturation might occur when the quadrotor is flown under other conditions such as higher velocities and the model does not model the resulting behaviour.

6) Number of data sets: For the identification and validation of the models a total of four data sets were used. It would be beneficial for the results if more data sets had been available.

For this research the emphasis was the validation of the model. Only one data set was therefore used for estimation per model and three for validation. However, it can be seen that the standard deviation in the fit percentages is still quite large. To decrease the standard deviation in the validation scores in table II more validation data sets can be used.

Secondly, another approach that could be tried in further research is using multiple data sets for the estimation of the model. This would most likely decrease the overfitting and thus decrease the standard deviation in the fit scores. With more data sets available there would still be enough data sets left to validate the model over.

E. Simulation

When looking at the simulation result, it stands out that the fit percentages for the input response models are a lot higher then for the dynamics models. This can be seen when comparing the response for ϕ, θ and v_z in Fig. 9 to the response in v_x and v_y in Fig. 6. Since the dynamics models are derived analytically, this difference is probably a result of the assumption that the aerodynamic drag is linear with the velocity.

VII. Conclusion

In the introduction, the research goal is stated as: *Derive a model that describes the dynamics of the Bebop 2 drone accurately*. In this report a first order model, a first order model plus time delay, a second order model, a third order model and a fourth order model have been derived and estimated. The third order model is the most accurate with a NRMSE fit percentage of $83.5 \pm 1.6\%$. However, the difference in the fit percentage between the second and third order model is only 0.6 percentage point. Since the accuracy is comparable, the complexity of the model will most likely be decisive for which model is preferable for a specific application.

Although the goal of this research has been achieved, there are some aspects that could be improved upon and added in further research.

First of all, to decrease the variance in the estimated parameters, longer inputs can be used for each data set. Another option is to record more data sets.

Secondly, a central composite design could be conducted for the second and higher order models to minimize the chances of converging to a local minimum when performing the parameter estimation.

Lastly, to improve the model further research on the yaw identification could be done.

Acknowledgements

In completing our Bachelor End Project, we received help and guidance of multiple individuals who deserve our sincere gratitude. We would like to extend a special thank you towards H. Zhu for giving us constructive feedback, guidance and assignments throughout numerous meetings. We would also like to thank him for providing the necessary software to complete this project. In addition, we would like to thank Dr. J. Alonso Mora for advise and feedback on various aspects of the project. We also want to thank the Delft Center for Systems and Control for allowing us unlimited access to the lab and the Motion Capture System in particular. Lastly, we thank the students and staff of the Cognitive Robotics department for their fresh insights and feedback during the colloquia.

References

[1] Dhl drone delivery and parcelcopter technology. Retrieved from: https://discover.dhl.com/business/business-ethics/parcelcopter-drone-technology. Online; accessed 27-05-2019.

[2] A Momont. Ambulance drone. Retrieved from: https://www.tudelft.nl/en/ide/research/research-labs/applied-labs/ambulance-drone/. Online; accessed 27-05-2019.

[3] Parrot. Parrot bebop 2. Retreived from: https://www.parrot.com/us/drones/parrot-bebop-2. Online; accessed 01-06-2019.

[4] R. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation*, 34(3):276–280, March 1986.

[5] Ionel Stanculeanu and Theodor Borangiu. Quadrotor black-box system identification. 5:276–279, 06 2011.

[6] Ross E. Allen and Marco Pavone. A real-time framework for kinodynamic planning in dynamic environments with application to quadrotor obstacle avoidance. *Robotics and Autonomous Systems*, 115:174 – 193, 2019.

[7] En-Hui Zheng, Jing-Jing Xiong, and Ji-Liang Luo. Second order sliding mode control for a quadrotor uav. *ISA Transactions*, 53(4):1350 – 1356, 2014. Disturbance Estimation and Mitigation.

[8] I. Morar and I. Nascu. Model simplification of an unmanned aerial vehicle. In *Proceedings of 2012 IEEE International Conference on Automation, Quality and Testing, Robotics*, pages 591–596, May 2012.

[9] M. Hehn and R. D'Andrea. Quadrocopter trajectory generation and control. volume 44, pages 1485–1491, 2011.

[10] Yi-Rui Tang, Xiao Xiao, and Yangmin Li. Nonlinear dynamic modeling and hybrid control design with dynamic compensator for a small-scale uav quadrotor. *Measurement*, 109:51 – 64, 2017.

[11] T. Ngeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges. Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. *IEEE Robotics and Automation Letters*, 2(3):1696–1703, July 2017.

[12] Parrot. *ardroneSDK3*. Parrot, 1 edition, 1 2019. Accessed: 01-05-2019.

[13] Optitrack motion capture system. Retrieved from: https://optitrack.com/products/motive/tracker. Accessed: 21-04-2019.

[14] Robot operating system. Retrieved from: https:https://www.ros.org. Accessed: 21-04-2019.

[15] Mani Monajjemi. Ros driver for parrot bebop drone. https://bebop-autonomy.readthedocs.io/en/latest/ accessed: 26-03-2019.

[16] The Penssylvania State University. Central composite designs. Retreived from: https://newonlinecourses.science.psu.edu/stat503/node/59/. Online; accessed 18-05-2019.

[17] Babatunde Olawoye. *a comprehensive handout on central composite design*. 07 2016.

[18] Matlab Documentation. Simulate and predict identified model output. Retrieved from: https://nl.mathworks.com/help/ident/ug/definition-simulation-and-prediction.html. Online; accessed 24-05-2019.

[19] N.Potdar. Online trajectory planning and control of a mav payload system in dynamic environments. April 2018.

[20] R. L. Payne. Optimal experiment design for dynamics system identification, 1974.

[21] P.M.J. Van den Hof. System identification - data-driven modelling of dynamic systems. Retreived from: http://www.dcsc.tudelft.nl/ discsysid/-ManuscrSysid_Feb2012p̊df. Online; accessed 03-05-2019.

[22] Necessary bandwidth to measure a digital signal with a specific rise time. Retrieved from: https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8qGSAS&l=nl-NL. Online; accessed 27-05-2019.

[23] Eero Simoncelli. Least squares optimization. *Lecture Notes, http://www. cns. nyu. edu/eero/teaching. html*, 01 2003.

[24] Henri P. Gavin. The levenberg-marquardt method for nonlinear least squares curve-fitting problems c . 2019.

[25] Manolis Lourakis. A brief description of the levenberg-marquardt algorithm implemened by levmar. *A Brief Description of the Levenberg-Marquardt Algorithm Implemented by Levmar*, 4, 01 2005.

[26] Ananth Ranganathan. The levenberg-marquardt algorithm. 2004.

[27] Minitab Support. What are response surface designs, central composite designs, and box-behnken designs? Online; accessed 21-05-2019.

[28] Matlab Documentation. Idnlgreymatlab. Retreived from: https://nl.mathworkscom/help/ident/ref/idnlgrey.html?searchHighlight=idnlgrey&s_tid=doc_srchtitle. Online; accessed 24-05-2019.

[29] B. Southall, B. F. Buxton, and J. A. Marchant. Controllability and observability: Tools for kalman filter design. Proceedings of the British Machine Vision Conference. 1998.

APPENDIX A
ESTIMATION OF k_{D_z}

The drag coefficient in the z-direction is not taken into account in the quadrotor dynamics equations in section II-B. This is due to several reasons:

Firstly, the dynamics in the z-direction are directly controlled by the internal attitude controller. This can be seen in the input vector $\mathbf{u}$ where v_z is an input for the quadrotor. The drag in the z-direction is thus modelled by the proposed model for the vertical velocity.

Secondly, the vertical velocity term in v_x and v_y is hard to determine. This can be seen by evaluating Eq. (4) assuming the drag coefficient in the z-direction is taken into account in Eq. (19).

$$\begin{cases} \dot{v}_x = k^*_{D_y}\sin(\psi)(v_y\cos(\psi) - v_x\sin(\psi)) - k^*_{D_x}\cos(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) + (\cos(\psi)\frac{\tan(\theta)}{cos(\phi)} + \tan(\varphi)\sin(\psi))(\dot{v_z} + g + k^*_{D_z}v_z) \\ \dot{v}_y = -k^*_{D_x}\sin(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) - k^*_{D_y}\cos(\psi)(v_y\cos(\psi) - v_x\sin(\psi))(\sin(\psi)\frac{\tan(\theta)}{cos(\phi)} - \cos(\psi)\tan(\varphi))(\dot{v_z} + g + k^*_{D_z}v_z) \end{cases} \tag{19}$$

It is shown that a k_{D_z} term is added at the end of $\dot{v}_x$ and $\dot{v}_y$ equations. This reveals the relationship between the horizontal accelerations and the vertical velocity. The k_{D_z} only has an effect on $\dot{v}_x$ and $\dot{v}_y$ when there is a v_z present. This makes it very hard to determine, since it requires simultaneous inputs in the x-, y- and z-direction.

APPENDIX B
FULL DYNAMICS MODELS

The full dynamics model of the drone is the result of solving the equation of motion of the system for the x and y components in the world frame. Since the vertical velocity is an input of the quadrotor, it is a known quantity. The equation of motion is given by Eq. (20).

$$m\dot{\mathbf{v}}^{\mathcal{A}} = -m\mathbf{g}^{\mathcal{A}} +^{\mathcal{A}} R_{\mathcal{B}}^{\mathcal{B}}\mathbf{T} - R_Z D {R_Z}^{-1}\mathbf{v}^{\mathcal{A}} \tag{20}$$

Evaluating the thrust as described in II-B yields:

$$T = \frac{m(\dot{v_z} + g)}{\cos(\varphi)\,\cos(\theta)} \tag{21}$$

Substituting this result in Eq. (20) gives the x and y components of the acceleration in the world frame.

$$\begin{cases} \dot{v_x} = k^*_{D_y}\sin(\psi)(v_y\cos(\psi) - v_x\sin(\psi)) - k^*_{D_x}\cos(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) + (\cos(\psi)\frac{\tan(\theta)}{cos(\phi)} + \tan(\varphi)\sin(\psi))(\dot{v_z} + g) \\ \dot{v_y} = -k^*_{D_x}\sin(\psi)(v_x\cos(\psi) + v_y\sin(\psi)) - k^*_{D_y}\cos(\psi)(v_y\cos(\psi) - v_x\sin(\psi)) + (\sin(\psi)\frac{\tan(\theta)}{cos(\phi)} - \cos(\psi)\tan(\varphi))(\dot{v_z} + g) \end{cases} \tag{22}$$

With $k^*_{D_{...}} = k_{D_{...}}/m$.

For the simulation of the quadrotor a full state space describing the dynamics is used. This state space differs per order model, so the models for a first order and general order model are given. For a model with time delay, the only change is a shift input, as illustrated in Eq. (16) and Eq. (18).

A. Full first order model

$$\mathbf{x} = \begin{bmatrix} x & y & z & v_x & v_y & v_z & \phi & \theta & \psi & \dot{\psi} \end{bmatrix}^T$$

$$\dot{\mathbf{x}} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \dot{v}_x \\ \dot{v}_y \\ \frac{1}{\tau_{v_z}}(k_{v_z}v_{zc} - v_z) \\ \frac{1}{\tau_\phi}(k_\phi\phi_c - \phi) \\ \frac{1}{\tau_\theta}(k_\theta\theta_c - \theta) \\ \dot{\psi} \\ \frac{1}{\tau_{\dot{\psi}}}(k_{\dot{\psi}}\dot{\psi}_c - \dot{\psi}) \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} x & y & z & v_x & v_y & v_z & \phi & \theta & \psi \end{bmatrix}^T$$

B. General order full model

$$\mathbf{x} = \begin{bmatrix} x & y & z & v_x & v_y & \mathbf{x}_{v_z} & \mathbf{x}_\phi & \mathbf{x}_\theta & \mathbf{x}_\psi & \psi \end{bmatrix}^T$$

$$\dot{\mathbf{x}} = \begin{bmatrix} v_x \\ v_y \\ C_{v_z}\mathbf{x}_{v_z} \\ \dot{v}_x \\ \dot{v}_y \\ A_{v_z}\mathbf{x}_{v_z} + B_{v_z}v_{zc} \\ A_\phi \mathbf{x}_\phi + B_\phi \phi_c \\ A_\theta \mathbf{x}_\theta + B_\theta \theta_c \\ A_\psi \mathbf{x}_\psi + B_\psi \dot{\psi}_c \\ C_\psi \mathbf{x}_\psi \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ C_{v_z}\mathbf{x}_{v_z} \\ C_\phi \mathbf{x}_\phi \\ C_\theta \mathbf{x}_\theta \\ \psi \end{bmatrix}$$

APPENDIX C
TIME DELAY

In figure 11, a step response on the pitch angle θ is plotted. The red dots represent the input, while the green and the blue line represent the first order and the first order plus time delay model. The grey line represents the measured data. The measured data indicates a present delay in the response of the drone. The percentage in the title gives the NRMSE fit for the models compared to the measured data. It is shown that the first order plus time delay estimates the measured data better then the model without time delay with an increase of 5% in the fit.

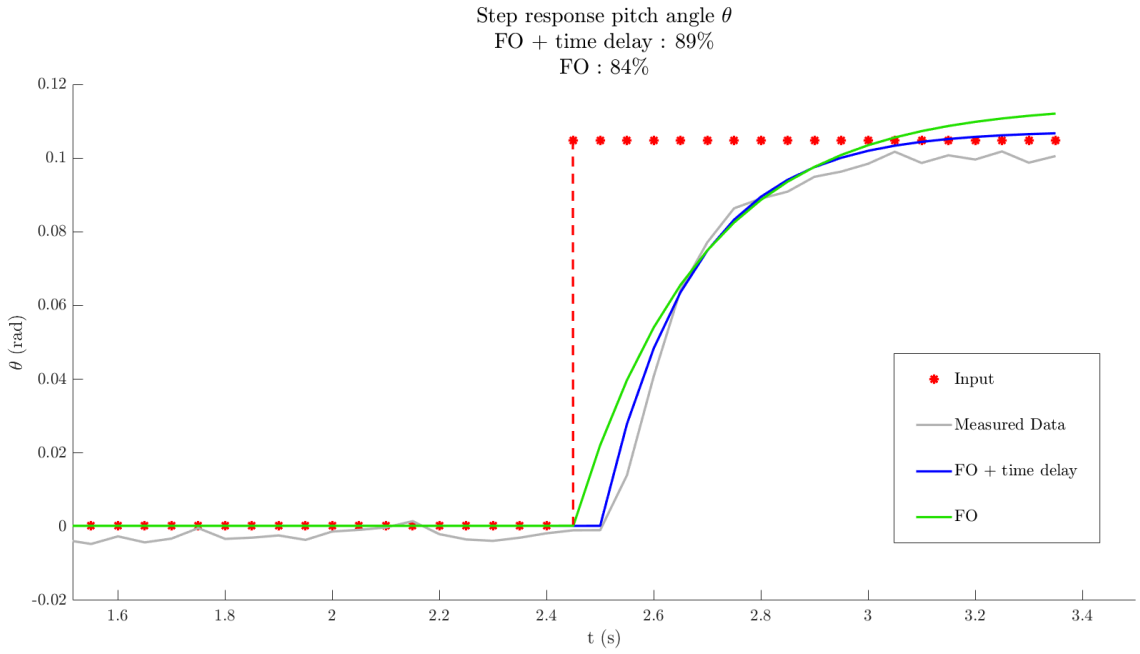

Fig. 11: Step response plot for the first order model with and without time-delay.

APPENDIX D
INPUT DESIGN

Designing input signals to use for system identification is a broad topic. There have been papers published on this topic finding different methods to generate input signals which would lead to optimal results according to certain criteria such as the variance of the parameter estimates [20]. However, because currently the real system of the quadrotor is only approximated by the model, designing input that excites the relevant frequency band suffices for the purposes of estimating an accurate motion model, as argued in [21].

A. Command frequency

A choice to be made in the designing of the input signal is at what frequency the command signal is to be send to the drone. In the SDK of the quadrotor it was found the inner control loop of the quadrotor runs at 20 Hz [12]. Therefore it is expected that there is no difference between sending commands at a lower rate than 20 Hz. However, sending commands faster than 20 Hz might be problematic because the quadrotor cannot process the commands that quickly and may buffer the execution. This was verified using a pitch step input signal. The input signal was send at different frequencies ranging from 5 Hz to 100 Hz.

If the quadrotor buffers commands that are sent at higher frequencies than its inner control loop, it is expected that for those command frequencies there is a longer delay, since the drone is still executing the zero input commands sent before the step input.

As shown in figure 12 the behaviour seems in line with this hypothesis. It can be seen that there is a larger delay in the response of the quadrotor when the commands are send at a frequency of higher than 20 Hz. Frequencies below 20 Hz produce nearly identical results. The data for each line is the average of three separate step response experiments at that frequency.

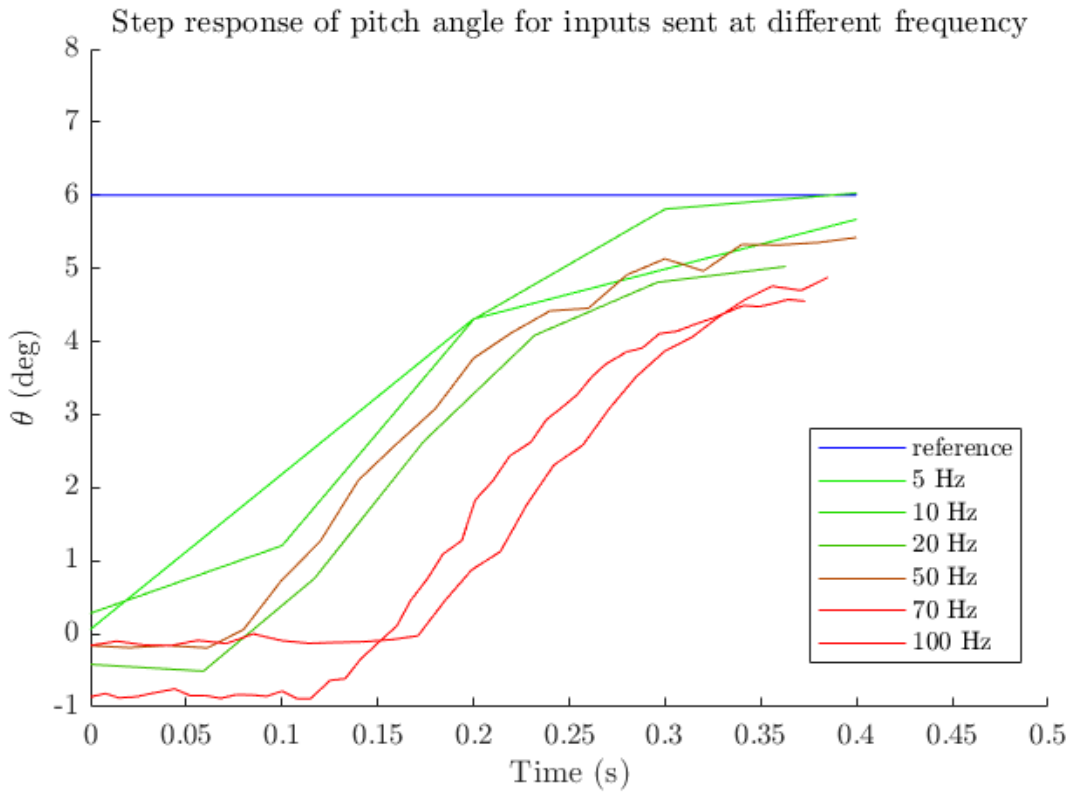

Fig. 12: Step response for different command frequencies

Because of these results all further experiments were conducted using a command frequency of 20 Hz. This also means that Mocap data is recorded at 20 Hz, since recording of the data happens in the same loop as sending the input commands.

B. System bandwidth

As mentioned before the input signal is to be designed to have the majority of its frequency spectrum within the system bandwidth. In order to calculate a reasonable bandwidth of the input signal, the system bandwidth is approximated from the

step response rise time. The system bandwidth approximately relates to the step response rise time by the following equation: [22]

$$bandwidth(Hz) = \frac{0.35}{risetime(s)} \tag{23}$$

To determine the rise time multiple experiments were conducted with step input signal on the pitch, roll and vertical velocity input channels separately. The inputs given were 12° for pitch and roll and 0.5 m/s for vertical velocity. The rise time was taken as the time it takes for the response to go from 10% to 90% of the steady state value. The results are shown in table III.

	Rise time step responses (s)		
	θ	ϕ	v_z
	0.301	0.350	0.300
	0.300	0.400	0.303
	0.200	0.450	0.300
	0.300	0.450	0.350
	0.200	0.398	-
avg $\pm$ σ	0.260 $\pm$0.055	0.410$\pm$0.0424	0.313$\pm$0.025

TABLE III: Rise time of step change of inputs

From the averages of these measured rise times the bandwidths of the responses for pitch, roll, and vertical velocity inputs were calculated. The standard deviations for the bandwidth are calculated using the standard formula for error propagation. [2]

The upper cutoff frequency of the input signal is set to the system bandwidth for every input. It is to be noted that the resulting system bandwidths are only approximations using Eq. (23) and only serve the purpose of being initial guesses towards the actual system bandwidth values.

The lower cutoff frequency follows from dimensions of the laboratory and was determined experimentally by trial and error. A lower cutoff frequency excites more frequencies but results in signals that are steady for longer periods of time which means the drone will eventually run out of space to move, after which an emergency stop must be activated. Table IV shows the cutoff frequencies for the different inputs. Figure 13 shows an example of the designed input for the pitch.

Input	Lower cutoff frequency (Hz)	Higher cutoff frequency (Hz)
θ	.30	1.35 ± 0.28
ϕ	.40	0.85 ± 0.08
v_z	.10	1.12 ± 0.09

TABLE IV: Cutoff frequencies for inputs

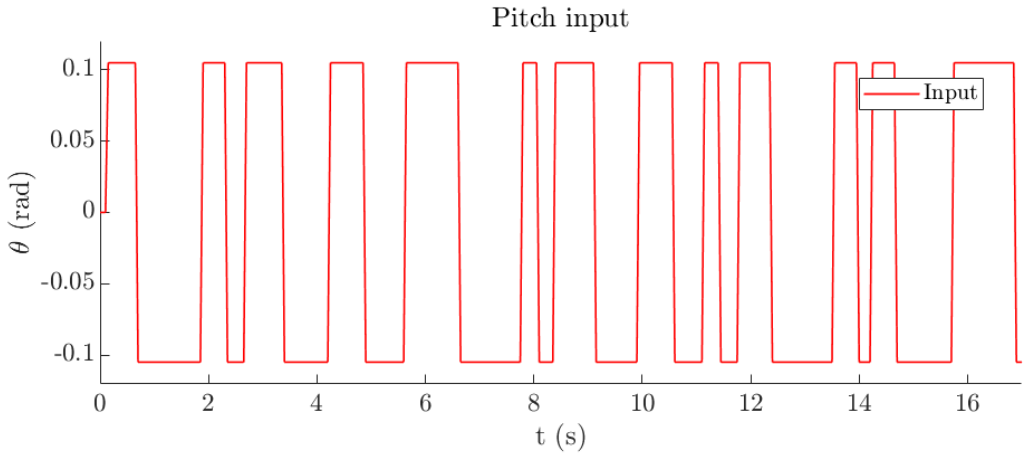

Fig. 13: Input example of RBS signal for pitch input

[2] $\sigma_{bw} = \sqrt{(\frac{\partial bw}{\partial rt})^2 \cdot \sigma_{rt}^2} = \frac{0.35}{rt^2} \cdot \sigma_{rt}$

APPENDIX E
INPUT DEPENDENCY

Models can be constructed in different ways. Firstly, it is possible to combine the results of single-input experiments into one model. During these experiments only one of the three inputs is excited with an RBS signal. For each input two of these experiments were conducted and the best model for that input was selected based on the validation. The estimated parameters for the three selected models are then used in one model.
Secondly, it is possible to estimate the model at once by using data from experiments where three inputs were excited with an RBS signal simultaneously[3]. Four of these experiments were conducted. For each experiment a model was estimated and validated using all four the data sets, resulting in a best all-input model.
Finally, the combined single-input model was also validated using the four all-input data sets and compared to the all-input model score.

To test whether there exists an input dependency, the model constructed with three single-input and the model constructed with the simultaneous input can be compared. Table V shows the fit percentages for these models of first order and first order plus time delay. It can be concluded that the percentages are sufficiently similar that there exists no dependency between inputs.

	All-input model	**Single-input Model**
1^{st} **(%)**	73.4±2.6	73.7±2.5
1^{st} **+TD (%)**	76.7±2.8	76.5±2.9

TABLE V: Lack of input dependency

[3]The signal for each input is not the same

APPENDIX F
YAW DRIFT

Figure 14 shows the drift noticed in the yaw direction when keeping the yaw rate input for the quadrotor at zero. With a zero input it is to be expected that the yaw angle does not change significantly because the quadrotor should remain stationary. However, it is clearly visible that during these experiments, the yaw drifts randomly before it converges to a steady state. The yawing happens mainly during and right after takeoff. All experiments were conducted with the same initial configuration and with the quadrotor being placed stationary on the ground.

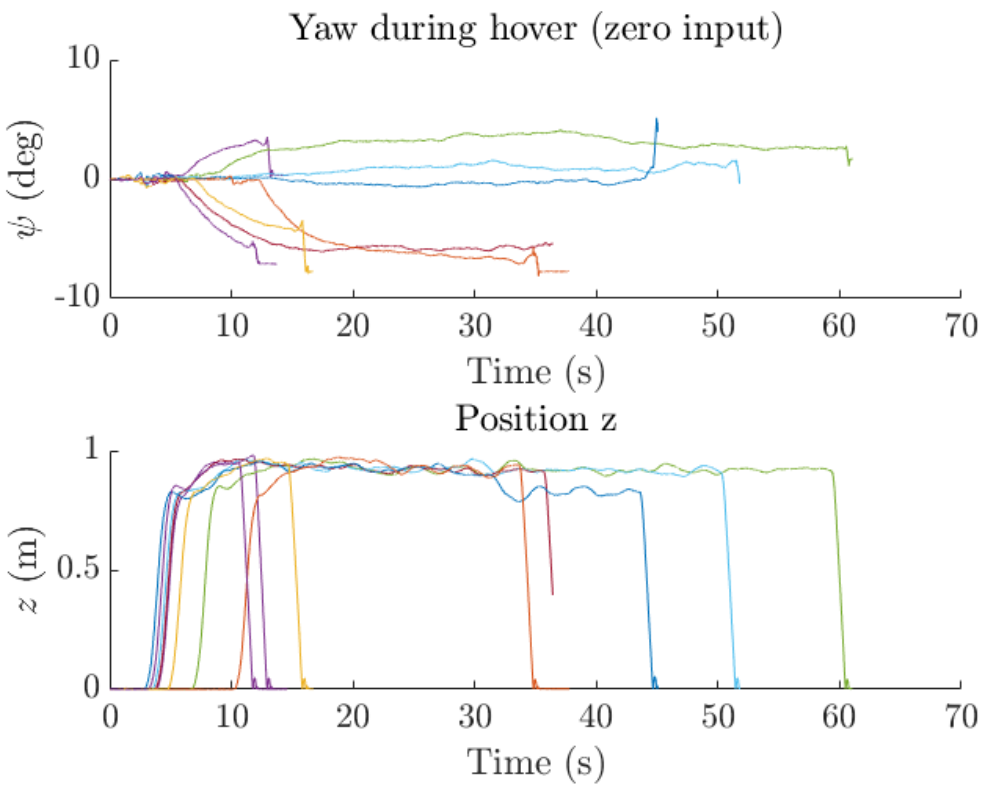

Fig. 14: Drift in yaw occurring during long experiments without input

Because of the randomness in the resulting yaw angle it was decided to derotate all data recorded in world frame, which includes the quadrotor's position and velocities, with the measured yaw angle. Performing this operation for experiments where the yaw input was kept at zero makes sure that any randomness in yaw angle is not considered in the system identification and thus does not influence that validity of the identified model.
Figure 15 shows the result of this derotation. In this particular experiment a step input was given to the pitch angle. This should only result in a velocity in the X-direction, assuming the quadrotor is placed sufficiently aligned with the world axes. However, it can be seen that a velocity in Y-direction also occurs, because the quadrotor yaws. Derotating the measured velocity minimizes this effect.

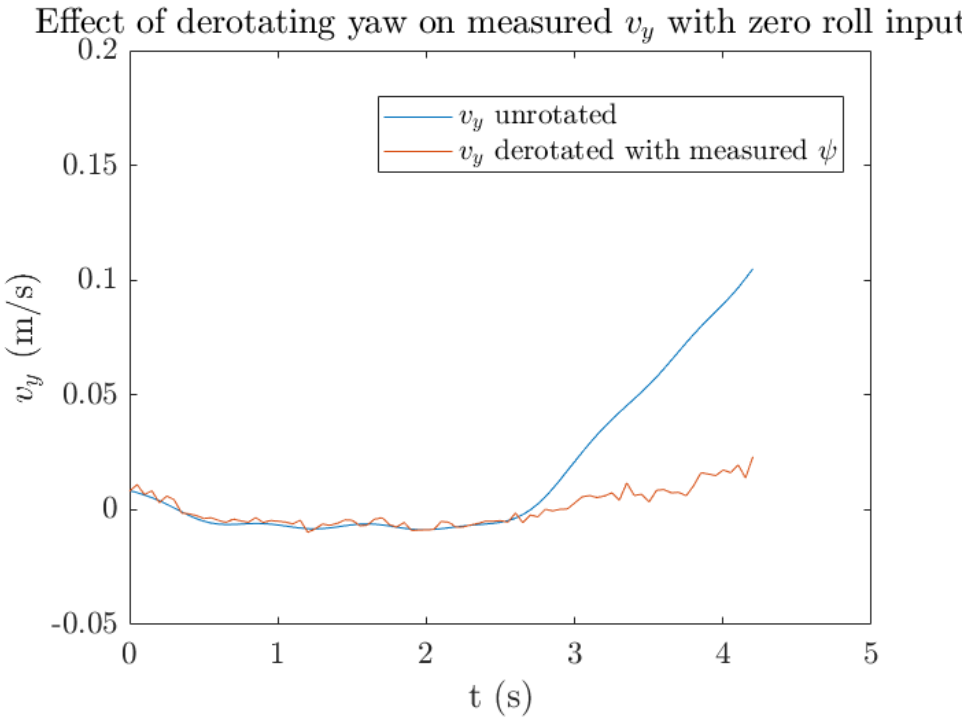

Fig. 15: Effect of derotating by yaw angle on irrelevant velocity. During this experiment, a step input on the pitch angle, only a velocity in X-direction is expected. Because of a drift in yaw a velocity in Y occurs as well. Derotating by the measured yaw angle minimizes this velocity component.

A possible explanation for this random drift before converging to steady state may have to do with the startup time of the individual rotors. If the rotor startup is not precisely controlled equally for each rotor yawing could occur when one rotor reaches a final velocity quicker than another rotor. In steady state the difference in added angular momentum would cancel out which would explain the converging of the yaw angle in steady state. It was noted during experiments that battery level may have an influence on the magnitude of yaw drifting occurring. However, investigating this effect is not within the scope of this research.

APPENDIX G
SYSTEM IDENTIFICATION

The level of system identification that is used is *grey box* system identification. This level is basically the fitting of a predetermined model to measured data by optimizing certain model parameters.
For this optimization, an objective function (Eq. (24)) is created. This function is the square of the difference between the measured and predicted data by the model (as shown in [23], [24] and [25]).

$$\chi = \sum_{i=0}^{n} (y_i - \hat{y}_i)^2 \tag{24}$$

In this equation y is the measured value of the output at data point i, and $\hat{y}$ the estimated value of the output. To be able to produce the best fit of the model to the measured data, this function has to be minimized. To find the minimum of this function, several numerical optimization methods are available. The three most popular methods are the steepest descent (gradient descent) method, the Gauss-Newton method and the Levenberg-Marquardt method. Since the Levenberg-Marquardt method converges quickly and is likely to converge to a minimum, this is the preferred method for optimization of the objective function χ. This algorithm combines the gradient descent method and the Gauss-Newton method [24]. These methods itself are not derived here, but can be found in [26] and [24]. This combination is made by adaptively switching between the gradient descent method and the Gauss-Newton method for the parameter updates. To determine this switching, a 'damping factor' λ is created. This λ is increased when an iteration results in an increase in the objective function χ. When an iteration results in a decrease in χ, λ is decreased. Small λ will result in the use of the Gauss-Newton method, which converged fairly fast, but can also converge to a local maximum instead of the desired minimum. Therefore, large values for λ result in the use of the gradient-descent method, which converges a lot slower, but will always converge to a local minimum. The result is a fairly quick minimization algorithm which will almost always converge to a minimum.

APPENDIX H
DESIGN OF EXPERIMENTS

For the optimization, a set of initial guesses has to be defined to reduce the chance of converging to a local instead of global minimum. This is done according to the Circumferential Central Composite Design. Figure 16 gives a graphical overview of this initial parameter choice for 3 parameters. The Central Composite Design generates 5 different evaluations per parameter (2 star points, 2 cube points and a center point). This spread is well suited for the fitting of models with quadratic behaviour according to Minitab [27]. Since it is not known what the least squares function will look like, this is a good first guess. The amount of points to be tried is given by the following formula:

$$N_{Points} = 2^k + 2 * k + N_c$$

where k is the amount of parameters that have to be estimated, and N_c is the number of center points. For the first order model described in section II-C1, there are 8 parameters which have to be estimated. Since a full black box response surface is not set up, the midpoint does not have to be evaluated multiple times. This results in $N_{Points} = 273$ for the first order model. The second order model described in section II-C3 has 26 parameters that have to be estimated. This would result in 67,108,917 initial points. Trying all these points would take up too much computational time.

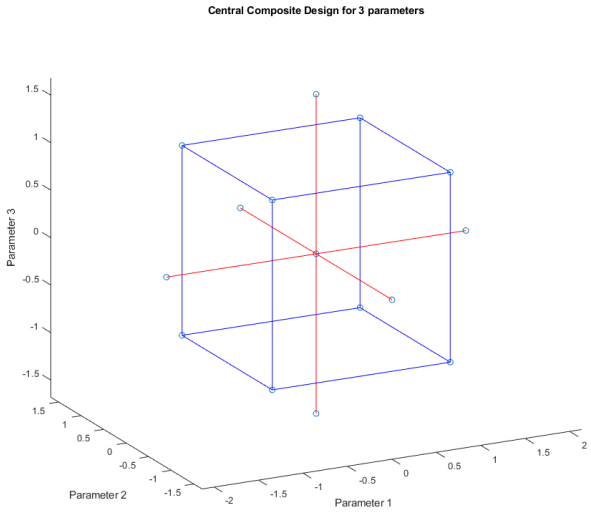

Fig. 16: Visual representation of central composite design

For higher order models, a random initial sample set is used. This is done by generating a random value for every parameter using `rand`. This thus generates a uniform distribution of points across a certain interval. If more time is available in future research the CCD method could be applied to ensure that the found models are indeed a global minimum.

APPENDIX I
INTEGRATION METHODS

To simulate the model response, the differential equations of the dynamics have to be solved. To do this, multiple numerical integration methods have to be solved. The `idnlgrey` model allows the following numerical integration methods according to the documentation [28]:

- ode45
  Runge-Kutta(4,5) solver.
- ode23
  Runge-Kutta(2,3) solver.
- ode113
  Adams-Bashforth-Moulton solver.
- ode15s
  Numerical differential formula solver.
- ode23s
  Modified Rosenbrock solver
- ode23t
  Trapezoidal solver
- ode23tb
  Implicit Runge-Kutta solver

To compare these methods, the same model is simulated using all these methods. The results can be seen in figure 17. Here, the response for the x position of a first order model is plotted. It can clearly be seen that the different integration methods have very little effect on the result. Since most methods lie within the range of less then centimeters for the x position, they are assumed almost equal.

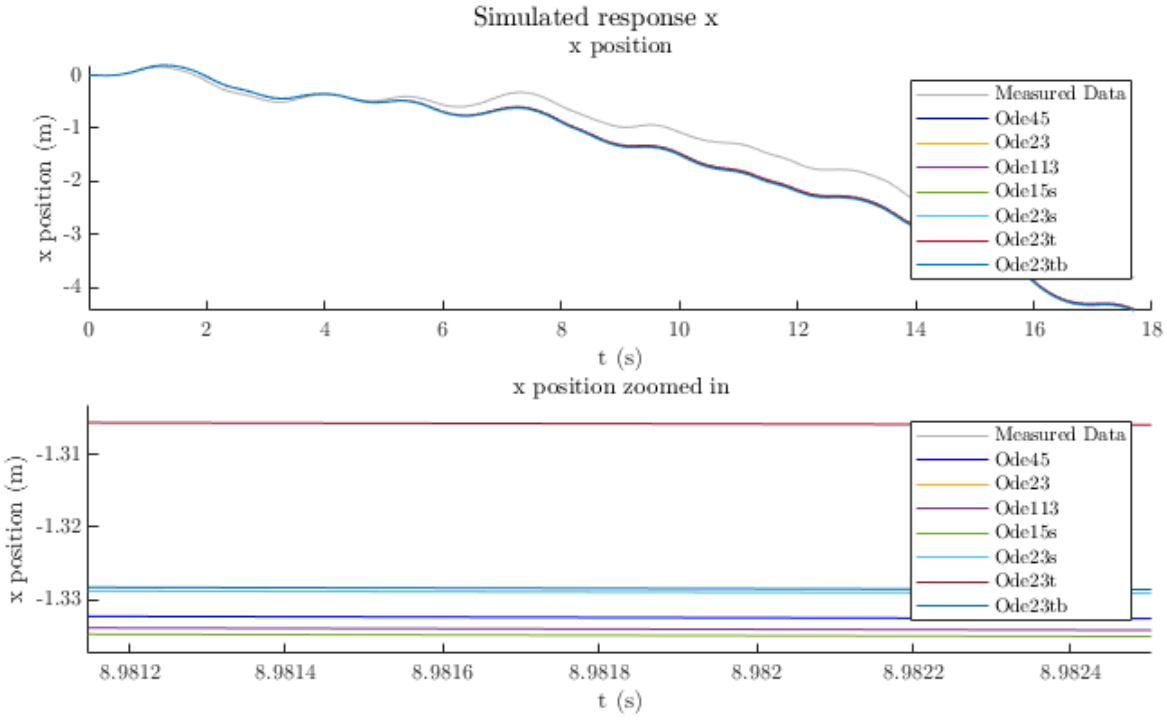

Fig. 17: Comparison between different integration methods.

APPENDIX J
YAW IDENTIFICATION

This section considers the yaw identification of the quadrotor. To complement the system identification a model is added to provide for the yaw identification. The yaw input differs from the pitch and roll input, since the yaw input consist of the yaw rate, rather then an absolute yaw angle. This is due to the fact that the yaw does not have a clear reference like the pitch and roll do. The model compares both the yaw rate input and output using the non-linear greybox model. The output is estimated with a Kalman filter as mentioned before.

The dynamics equation of a first order model plus time delay are shown. Furthermore in figure 18 the result of the identification is shown.

It must be noted that these results are computed with a data set that was not obtained in this research but provided by a third person. The results as shown are therefore only meant as example what the method would look like rather then results presented by the research.

$$\dot{\psi}(t) = \frac{1}{1.3542}(1.5979 \cdot \psi_c(t - 0.15) - \psi(t))$$

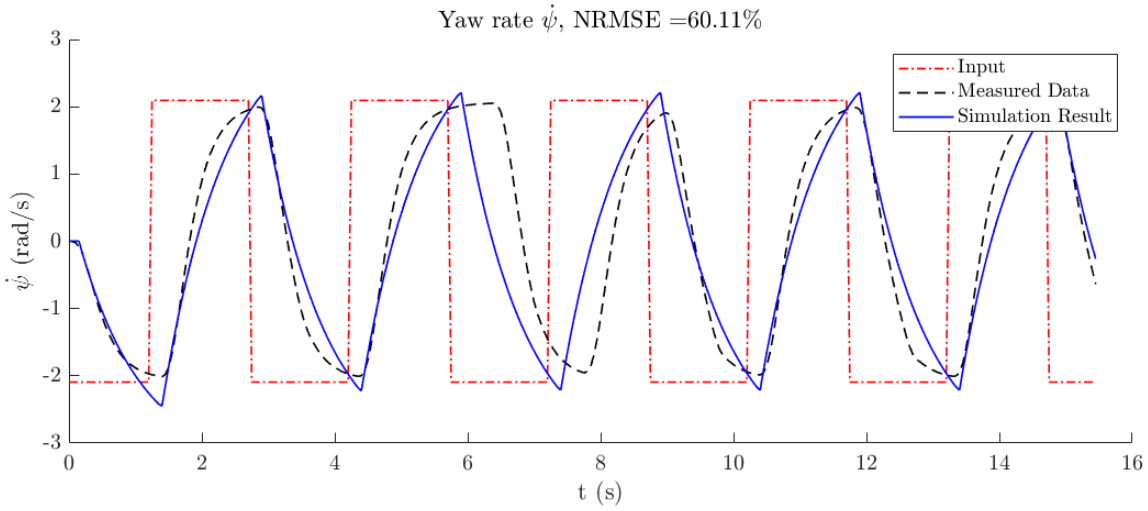

Fig. 18: Input response of first order plus time delay model for yaw

APPENDIX K
EXTENDED KALMAN FILTER

The states of the higher order models are not physical properties anymore (see Eq. (17)). Therefore, to perform the simulation the states need to be estimated. An Extended Kalman Filter (EKF) is used for this purpose. In order for an EKF to converge the system needs to be observable [29]. The observability matrix is thus as follows:

$$O = \begin{pmatrix} C \\ CA \\ CA^2 \\ CA^3 \\ CA^4 \\ CA^5 \end{pmatrix}$$

Where A and C are block diagonal matrices consisting of the identified dynamics and control matrices for ϕ, θ and v_z. For the system to be observable, the rank of O must be equal to the number of state variables. As an example, for the second order model the number of state variables of the input response model is 6. When matrix O is computed in MATLAB for the identified second order models and using the function `rank` this is true. For the EKF a state transition function and a measurement function are needed. The state transition function returns the state at $t = k$ given the state at $t = k - 1$ and is given by:

$$x[k] = x[k-1] + (Ax[k-1] + Bu[k-1])dT$$

Where dT is the sample time. The measurement function returns the output measurement at $t = k$ given the state at $t = k$ and is given by:

$$y[k] = Cx[k]$$

The Matlab functions `extendeKalmanfilter`, `correct` and `predict` are used to estimate the states. For each time step, first `correct` is used followed by `predict`. `correct` uses $\hat{x}[k|k-1]$ (the estimated state at $t = k$ using outputs up to $t = k-1$) to return $\hat{x}[k|k]$, this is the estimated state on $t = k$ using outputs up to $t = k$. `predict` uses $\hat{x}[k|k]$ to return $\hat{x}[k+1|k]$ which is then used by `correct` again for the next time step. Fig. 19 shows an example of how the state estimation for ϕ converges to measured angle for a measured data set. By adding data equal to the initial ϕ, the state estimation has converged before the start of the data, enabling the simulation to start there as well. Because the filter works well enough for the desired purpose a Unscented Kalman Filter was not used.

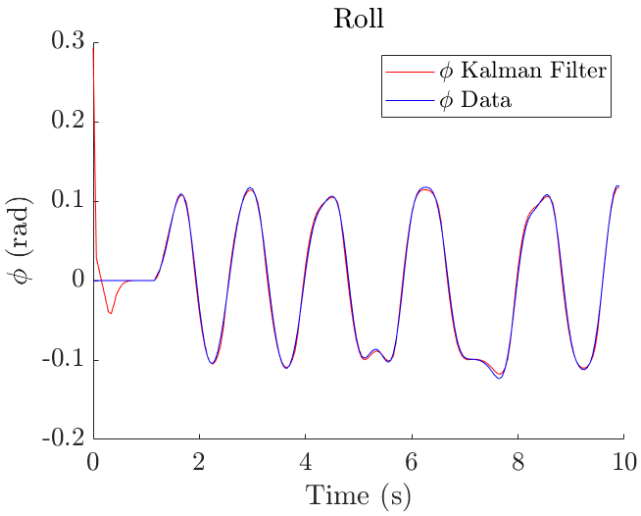

Fig. 19: State estimation for the roll angle using an Extended Kalman Filter

APPENDIX L
MODEL IDENTIFICATION RESULTS

In this section the results of the model identification for each type of model will be shown. The input response and the simulation of the drone are shown as well as the dynamics equation of the model. For all plots the input that is used is the input that the model fits best, but is not the data set the model was estimated on. Table VI shows the average fit percentages of all models. Adding a time delay to higher order models does not result in an improvement in the fit percentage.

Model order	1^{st}	1^{st} + TD	2^{nd}	2^{nd} + TD	3^{rd}	3^{rd} + TD	4^{th}	4^{th} + TD
fit (%)	73.4±2.6	77.8±2.8	82.9± 0.6	76.0± 0.7	83.5± 1.6	76.5± 0.8	82.3± 4.5	75.0± 2.8

TABLE VI: Fit scores for the best model of each type

A. First order model

The input response model for the best first order model is:

$$\dot{\phi} = \frac{1}{0.2368}(1.126 \cdot \phi_c - \phi)$$
$$\dot{\theta} = \frac{1}{0.2318}(1.1075 \cdot \theta_c - \theta)$$
$$\dot{v_z} = \frac{1}{0.3367}(1.2270 \cdot v_z - v_z)$$

The resulting plots and simulation are given in figures 20, 21 and 22.

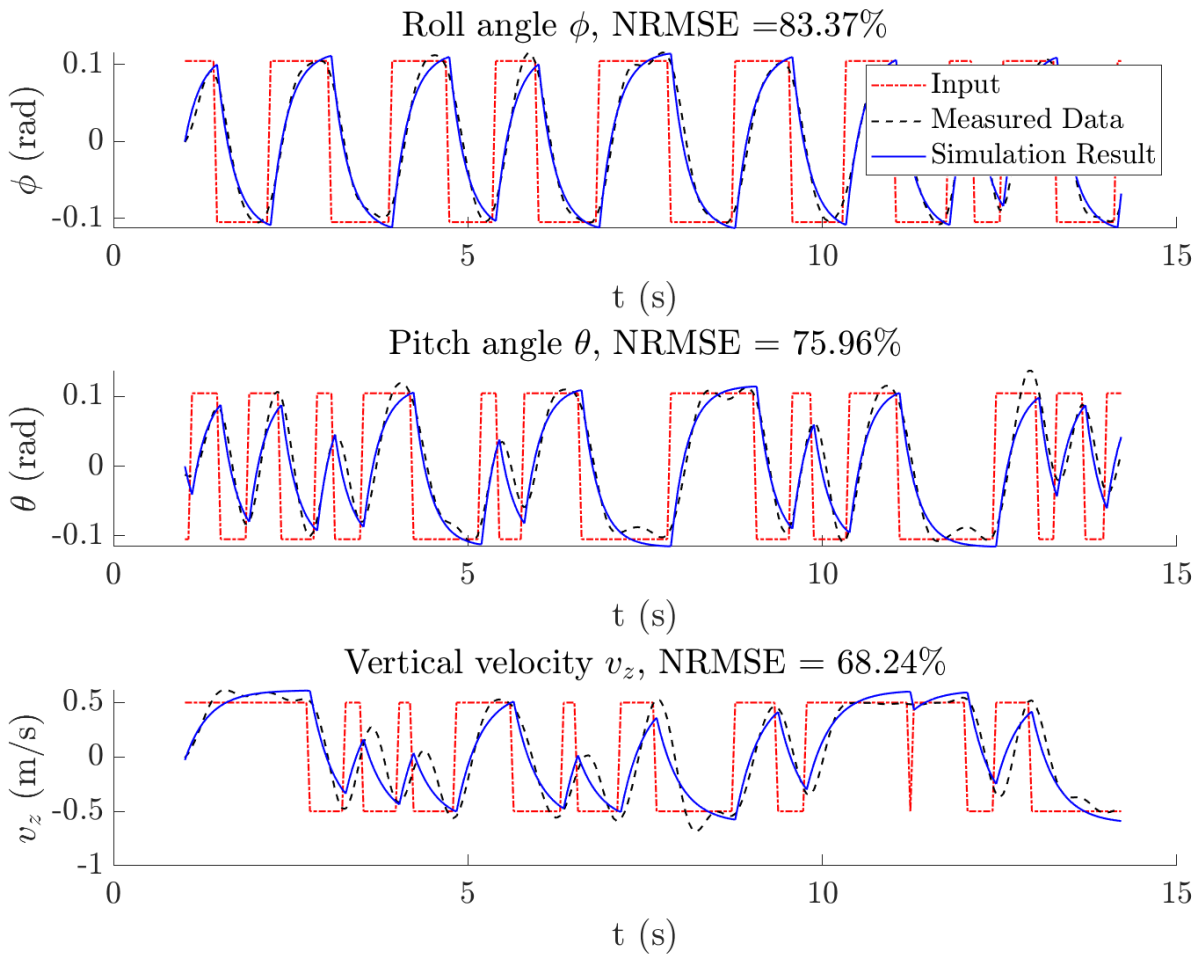

Fig. 20: Simulation results of input response for 1^{st} order model over best fitting data set

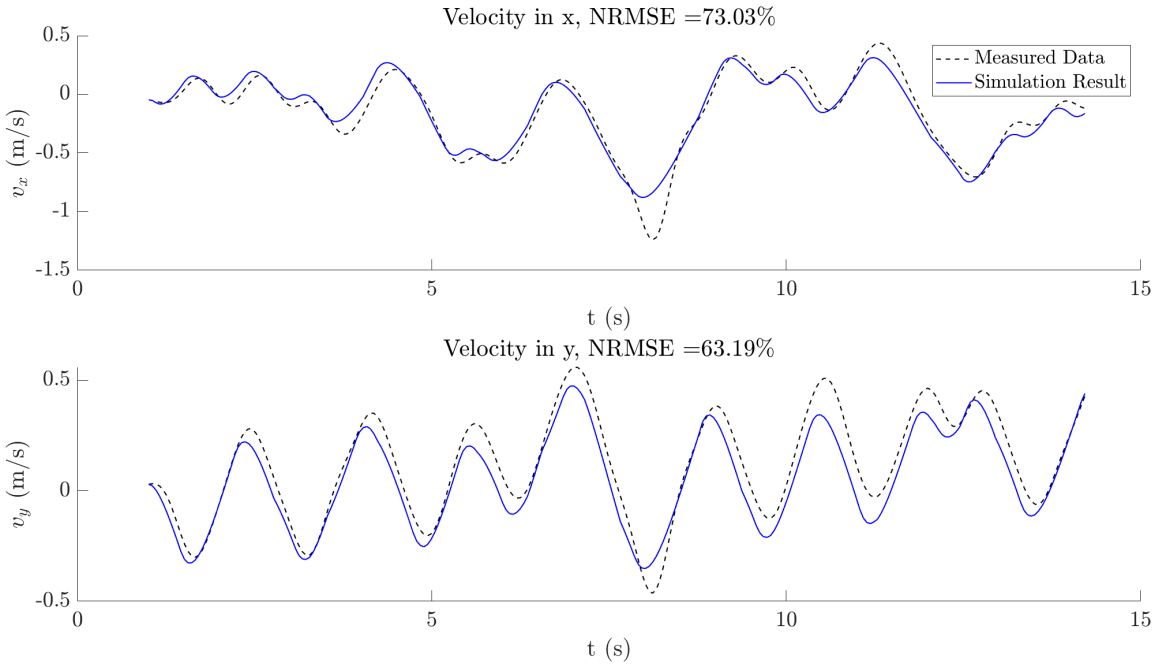

Fig. 21: Simulation results of velocity dynamics for 1^{st} order model over best fitting data set

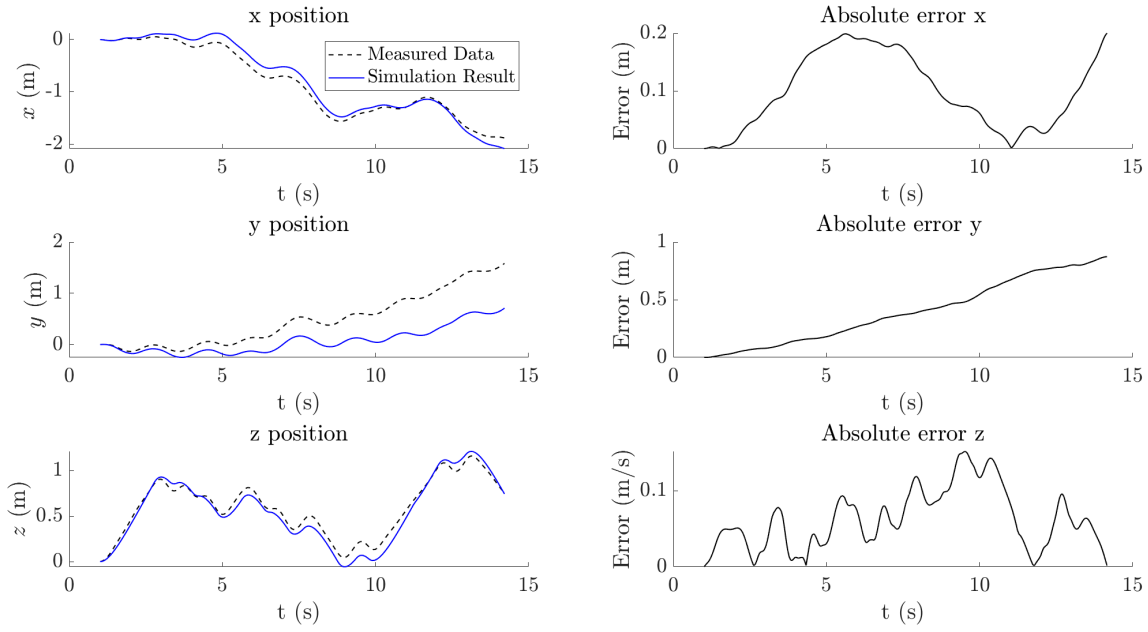

Fig. 22: Simulation results of position dynamics for 1^{st} order model over best fitting data set

B. First order plus time delay model

The input response model for the best first order plus time delay model is:

$$\dot{\phi}(t) = \frac{1}{0.1598}(0.9655 \cdot \phi_c(t-0.05) - \phi(t))$$
$$\dot{\theta}(t) = \frac{1}{0.1621}(0.9795 \cdot \theta_c(t-0.05) - \theta(t))$$
$$\dot{v}_z(t) = \frac{1}{0.2599}(1.1635 \cdot v_z(t-0.05) - v_z(t))$$

The resulting plots and simulation are given in figures 23, 24 and 25.

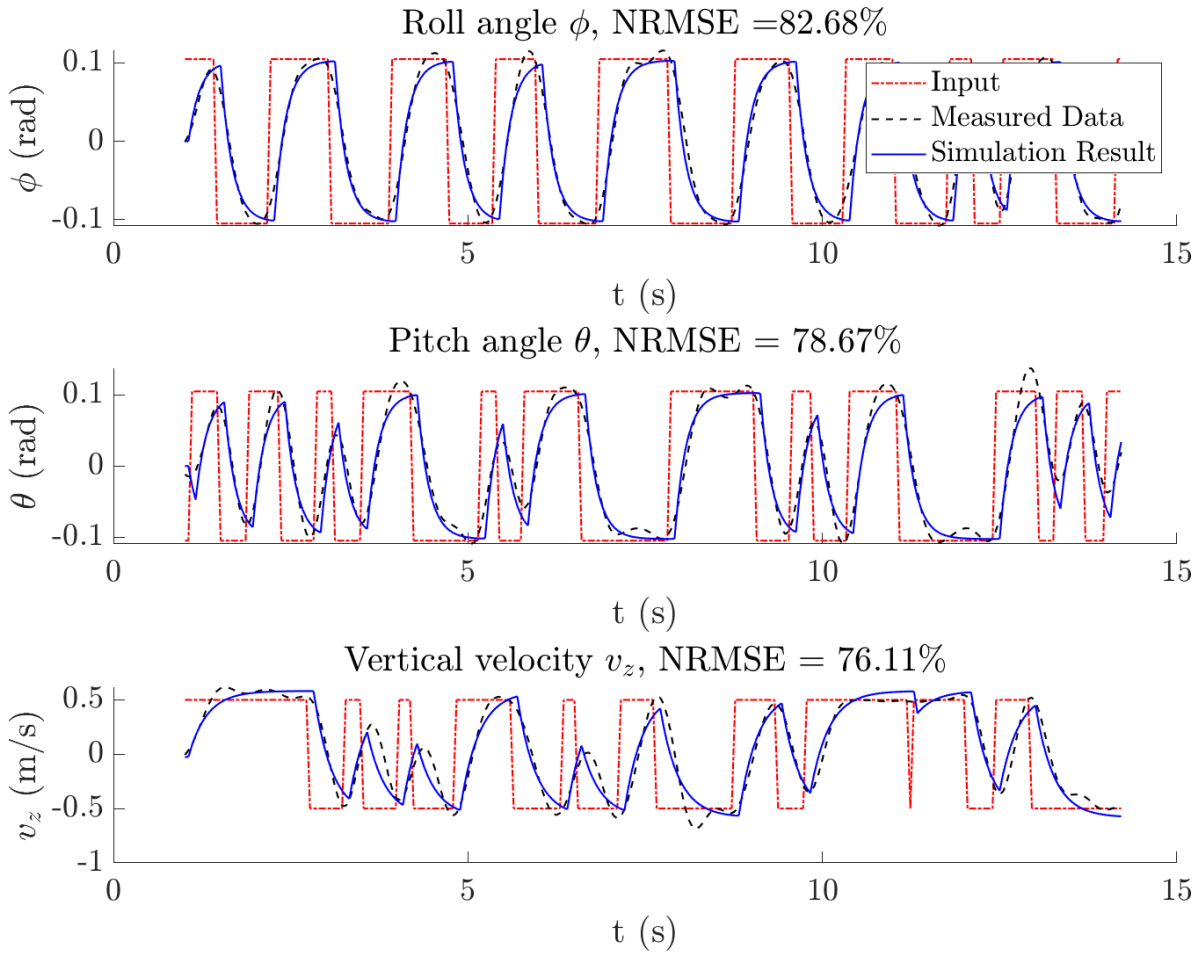

Fig. 23: Simulation results of input response for 1^{st} order + TD model over best fitting data set

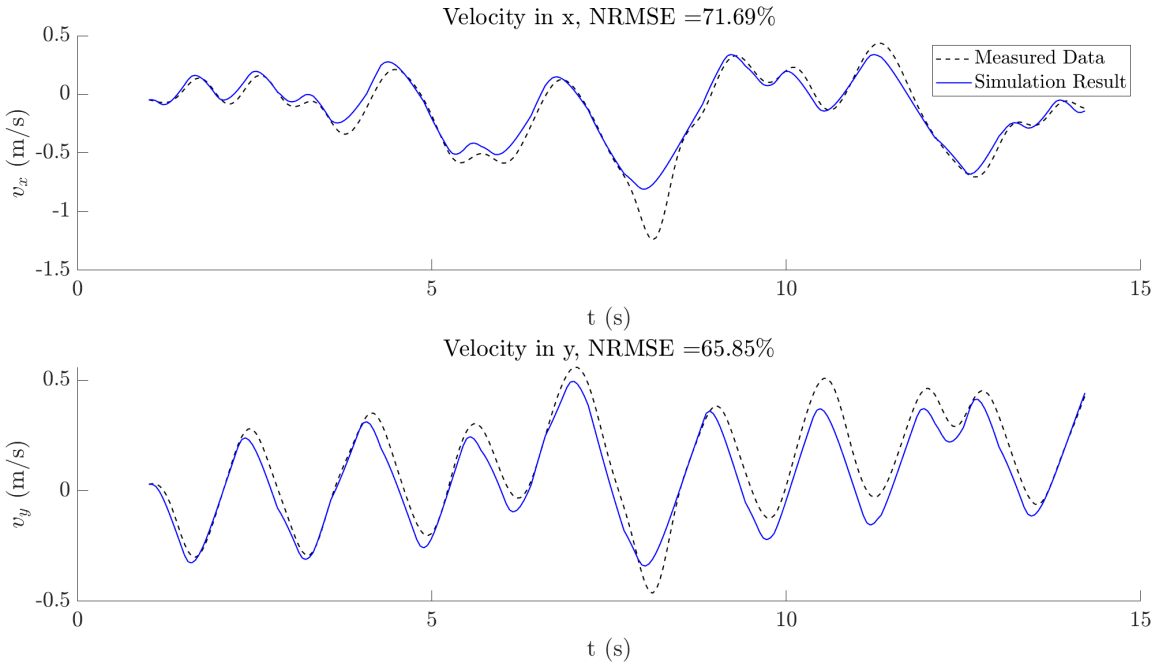

Fig. 24: Simulation results of velocity dynamics for 1^{st} order + TD model over best fitting data set

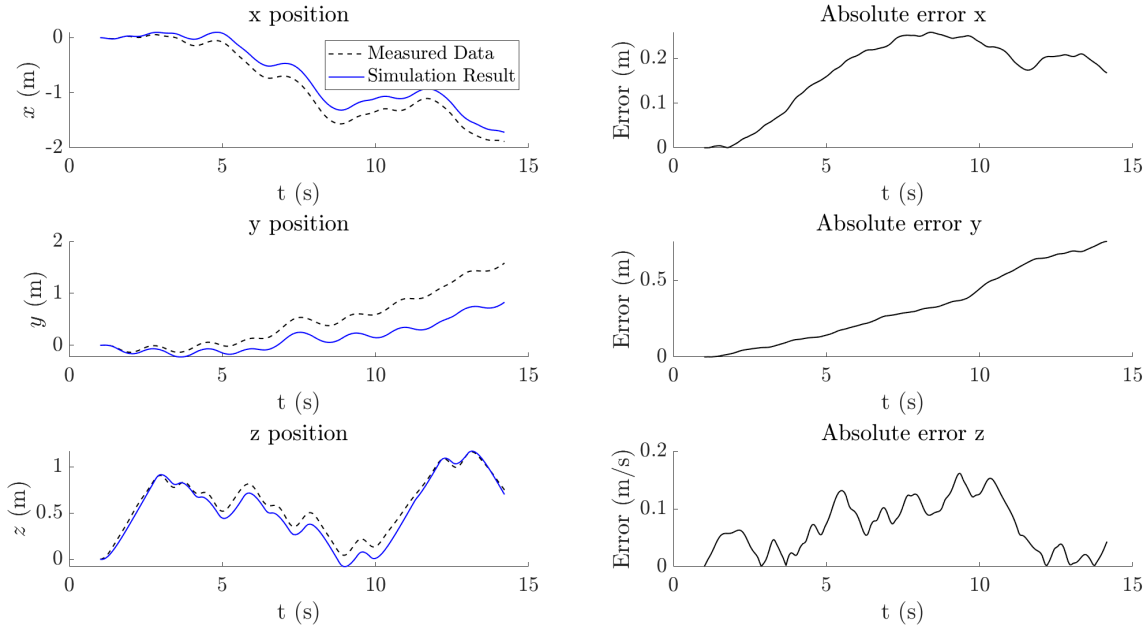

Fig. 25: Simulation results of position dynamics for 1^{st} order + TD model over best fitting data set

C. Second order model

$$\dot{\mathbf{x}}_\phi = \begin{bmatrix} -3.6268 & 4.5680 \\ -4.2580 & -3.8136 \end{bmatrix} \mathbf{x}_\phi + \begin{bmatrix} 3.3057 \\ -1.5443 \end{bmatrix} \phi_c$$

$$\dot{\mathbf{x}}_\theta = \begin{bmatrix} -6.5559 & 5.4182 \\ -4.8719 & -5.1334 \end{bmatrix} \mathbf{x}_\theta + \begin{bmatrix} -0.2825 \\ -3.9654 \end{bmatrix} \theta_c$$

$$\dot{\mathbf{x}}_{v_z} = \begin{bmatrix} -4.5702 & 4.4186 \\ -4.9402 & -4.7852 \end{bmatrix} \mathbf{x}_{v_z} + \begin{bmatrix} 3.3742 \\ 0.4442 \end{bmatrix} v_{z_c}$$

$$\phi = \begin{bmatrix} -0.1337 & -1.4979 \end{bmatrix} \mathbf{x}_\phi$$

$$\theta = \begin{bmatrix} -2.2622 & -0.1050 \end{bmatrix} \mathbf{x}_\theta$$

$$v_z = \begin{bmatrix} 0.1768 & -2.5312 \end{bmatrix} \mathbf{x}_{v_z}$$

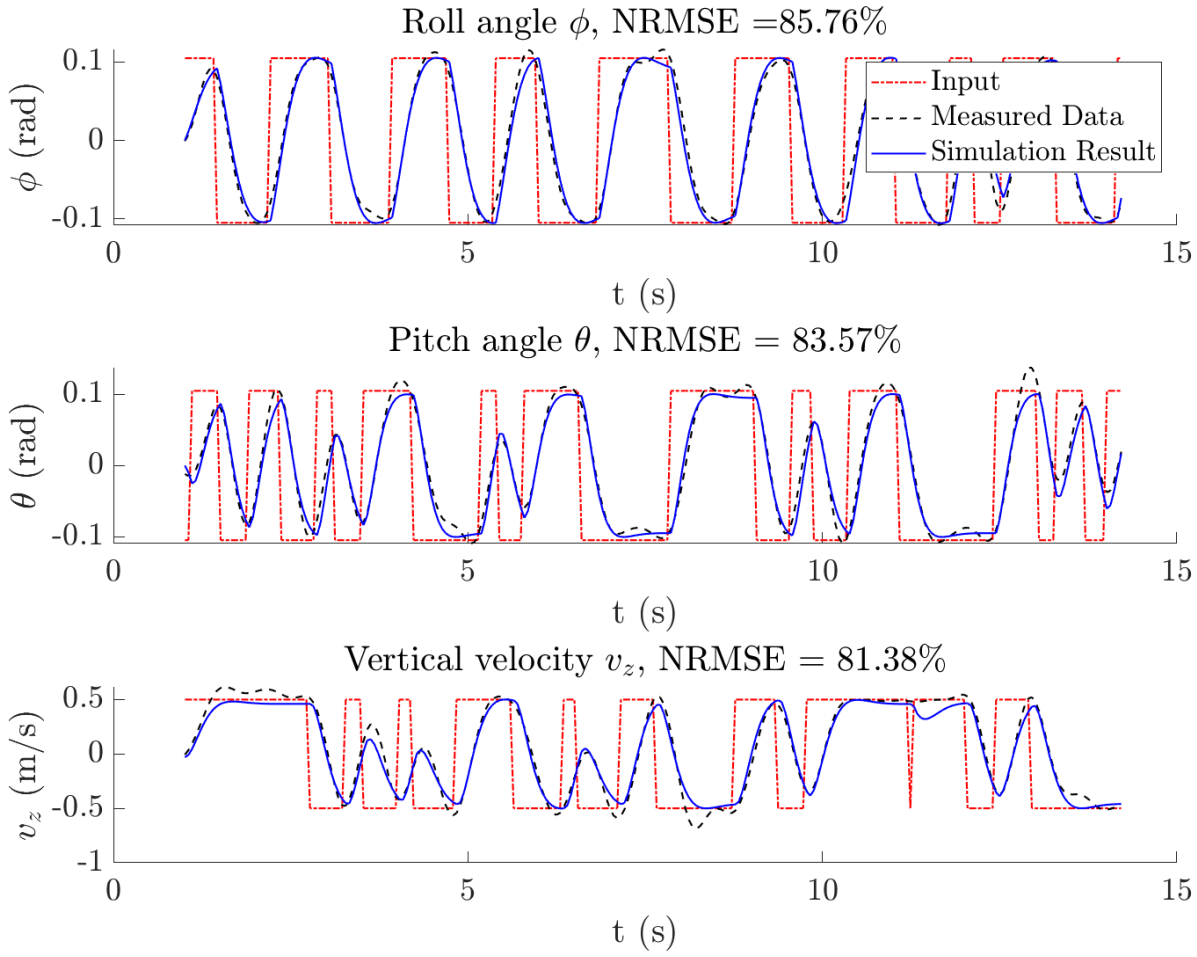

Fig. 26: Simulation results of input response for 2^{nd} order model over best fitting data set

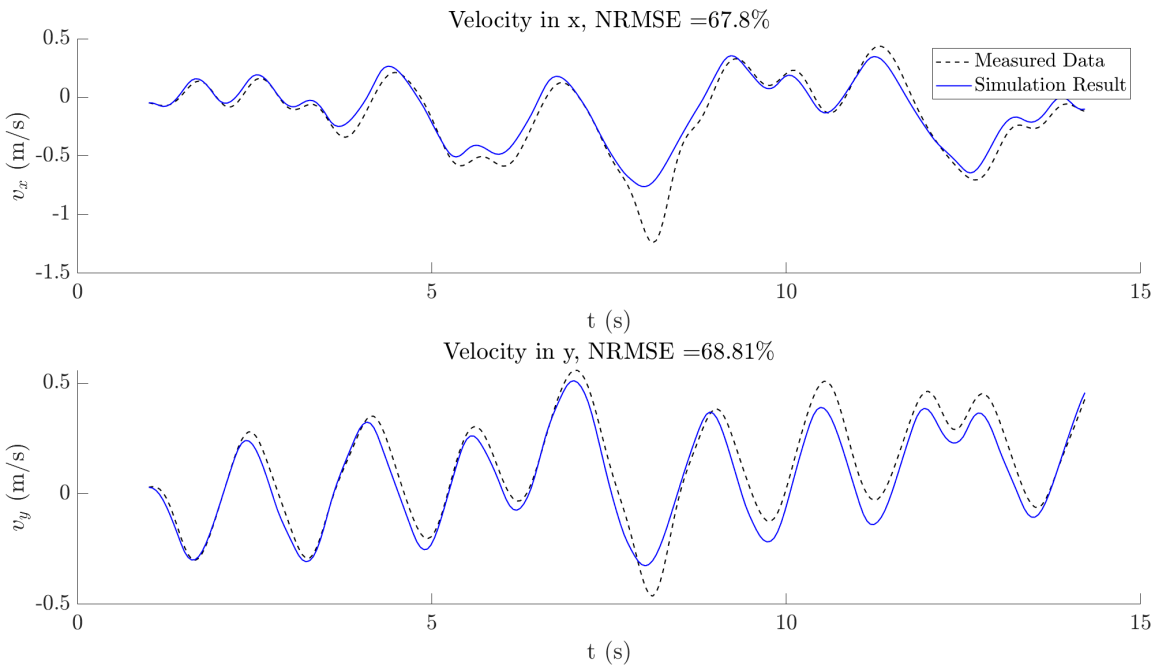

Fig. 27: Simulation results of velocity dynamics for 2^{nd} order model over best fitting data set

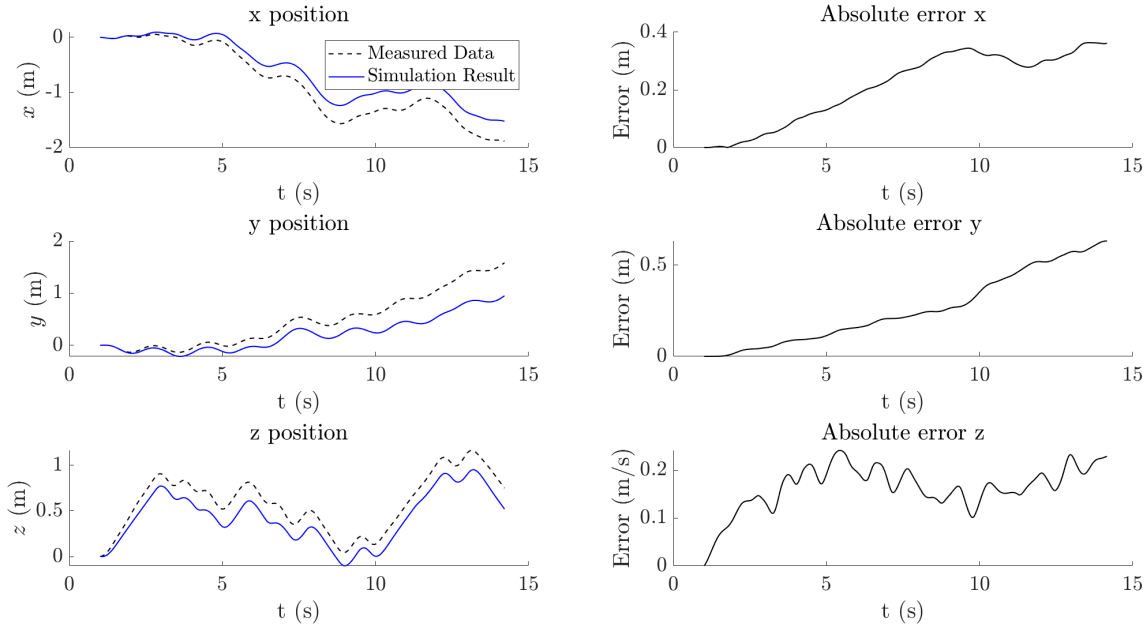

Fig. 28: Simulation results of position dynamics for 2^{nd} order model over best fitting data set

D. Third order model

$$\dot{\mathbf{x}}_\phi = \begin{bmatrix} -3.0696 & -3.5650 & -2.4296 \\ 0.0520 & -5.4143 & -5.9011 \\ 3.7089 & 1.0430 & -1.4279 \end{bmatrix} \mathbf{x}_\phi + \begin{bmatrix} -2.1462 \\ 1.6727 \\ 3.5351 \end{bmatrix} \phi_c$$

$$\dot{\mathbf{x}}_\theta = \begin{bmatrix} -6.1824 & -1.8848 & -5.5527 \\ -2.7956 & -1.6305 & 2.0099 \\ 2.9196 & 2.5074 & -6.8438 \end{bmatrix} \mathbf{x}_\theta + \begin{bmatrix} 3.2526 \\ 1.1928 \\ -0.7547 \end{bmatrix} \theta_c$$

$$\dot{\mathbf{x}}_{v_z} = \begin{bmatrix} -4.4829 & 4.6997 & 8.1645 \\ -3.2938 & -5.6196 & 2.7175 \\ -7.0201 & -1.7885 & -3.5407 \end{bmatrix} \mathbf{x}_{v_z} + \begin{bmatrix} 1.4509 \\ -3.2656 \\ 3.3287 \end{bmatrix} v_{z_c}$$

$$\phi = \begin{bmatrix} -0.5915 & -0.8319 & 0.4358 \end{bmatrix} \mathbf{x}_\phi$$

$$\theta = \begin{bmatrix} 1.5283 & -0.8466 & 3.8820 \end{bmatrix} \mathbf{x}_\theta$$

$$v_z = \begin{bmatrix} 0.5327 & -2.8051 & -2.4470 \end{bmatrix} \mathbf{x}_{v_z}$$

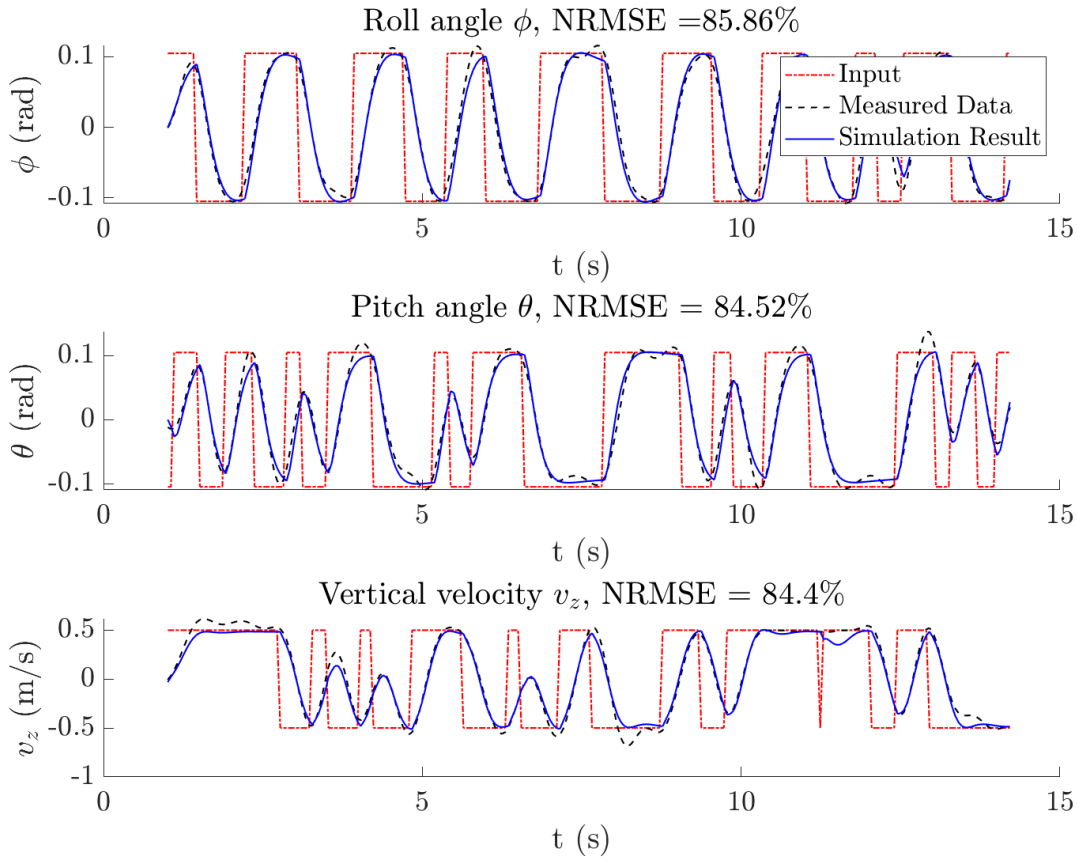

Fig. 29: Simulation results of input response for 3^{rd} order model over best fitting data set

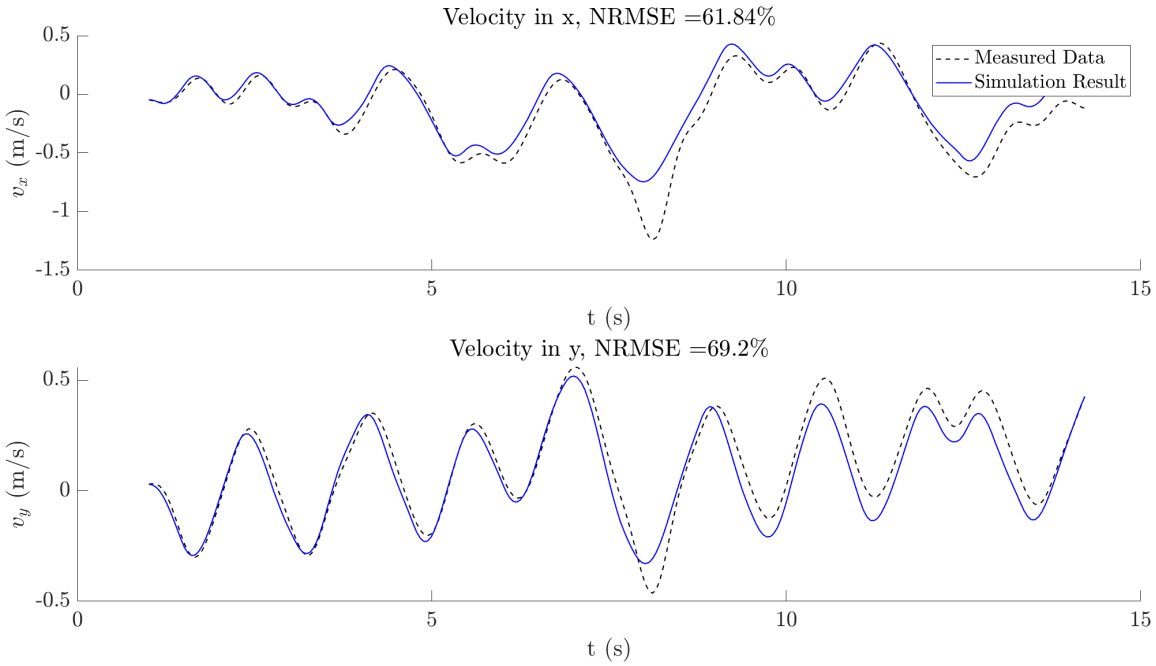

Fig. 30: Simulation results of velocity dynamics for 3^{rd} order model over best fitting data set

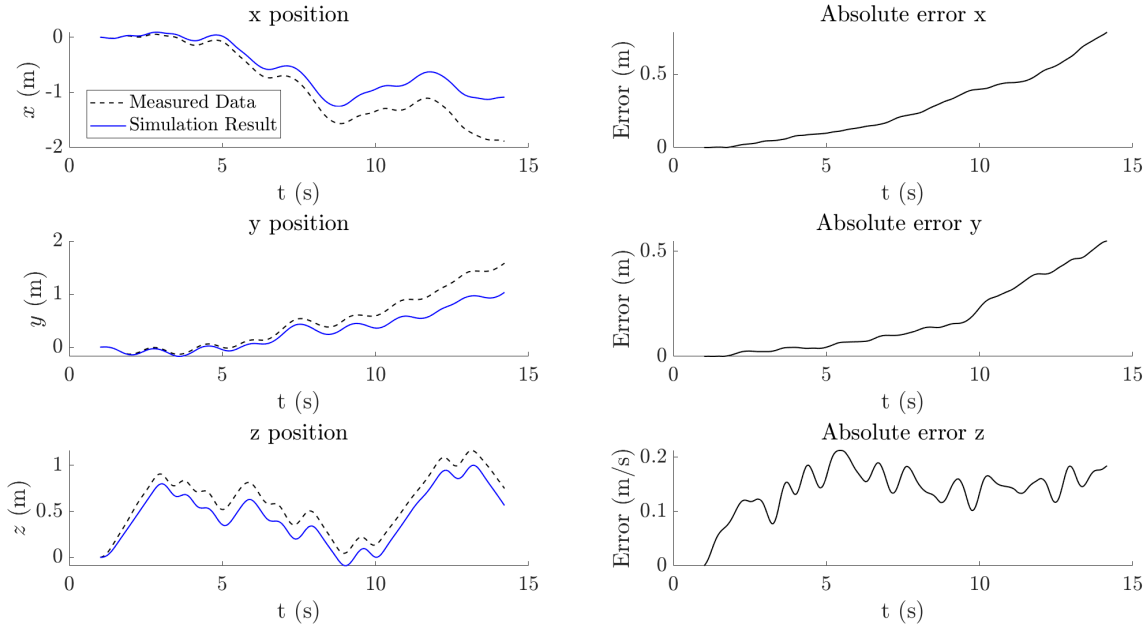

Fig. 31: Simulation results of position dynamics for 3^{rd} order model over best fitting data set

E. Fourth order model

$$\dot{\mathbf{x}}_\phi = \begin{bmatrix} -3.1466 & -2.2934 & 7.1880 & -1.5322 \\ 2.3268 & -4.8436 & -4.1068 & 2.6630 \\ -8.2756 & 3.8846 & -2.7173 & -5.1454 \\ 0.3868 & -5.3830 & 3.6690 & -0.8870 \end{bmatrix} \mathbf{x}_\phi + \begin{bmatrix} -0.8863 \\ -1.8831 \\ -2.0874 \\ -2.6259 \end{bmatrix} \phi_c$$

$$\dot{\mathbf{x}}_\theta = \begin{bmatrix} -1.6003 & -1.6104 & 2.5787 & 0.8336 \\ 0.5943 & -1.7293 & 1.4047 & 2.9910 \\ -2.1009 & 4.1525 & -4.5706 & -3.7690 \\ 0.4700 & -3.0666 & 5.0265 & -0.7570 \end{bmatrix} \mathbf{x}_\theta + \begin{bmatrix} 0.5930 \\ 1.3859 \\ -3.3685 \\ -0.0717 \end{bmatrix} \theta_c$$

$$\dot{\mathbf{x}}_{v_z} = \begin{bmatrix} -4.2459 & 5.8245 & 3.0821 & 2.5406 \\ -7.6343 & -2.7824 & 1.5316 & 1.5558 \\ -2.1061 & -4.1226 & -2.7998 & 2.0276 \\ -0.8159 & -1.0617 & 1.3522 & -7.2796 \end{bmatrix} \mathbf{x}_{v_z} + \begin{bmatrix} -2.5545 \\ -0.3099 \\ -3.5680 \\ -2.7811 \end{bmatrix} v_{z_c}$$

$$\phi = \begin{bmatrix} 2.8228 & 0.0985 & -0.0256 & -2.3801 \end{bmatrix} \mathbf{x}_\phi$$

$$\theta = \begin{bmatrix} 1.2238 & 0.4942 & -0.2987 & -1.6873 \end{bmatrix} \mathbf{x}_\theta$$

$$v_z = \begin{bmatrix} 0.2513 & 2.2975 & 1.6506 & -3.1097 \end{bmatrix} \mathbf{x}_{v_z}$$

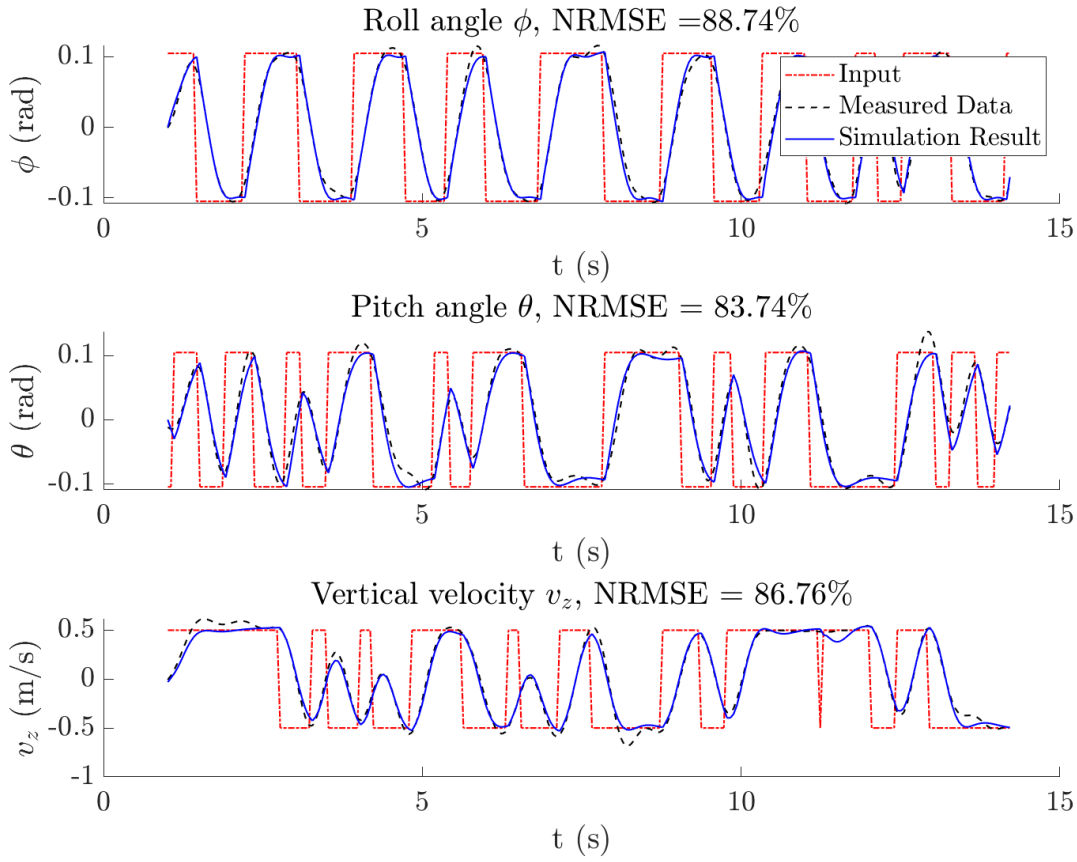

Fig. 32: Simulation results of input response for 4^{th} order model over best fitting data set

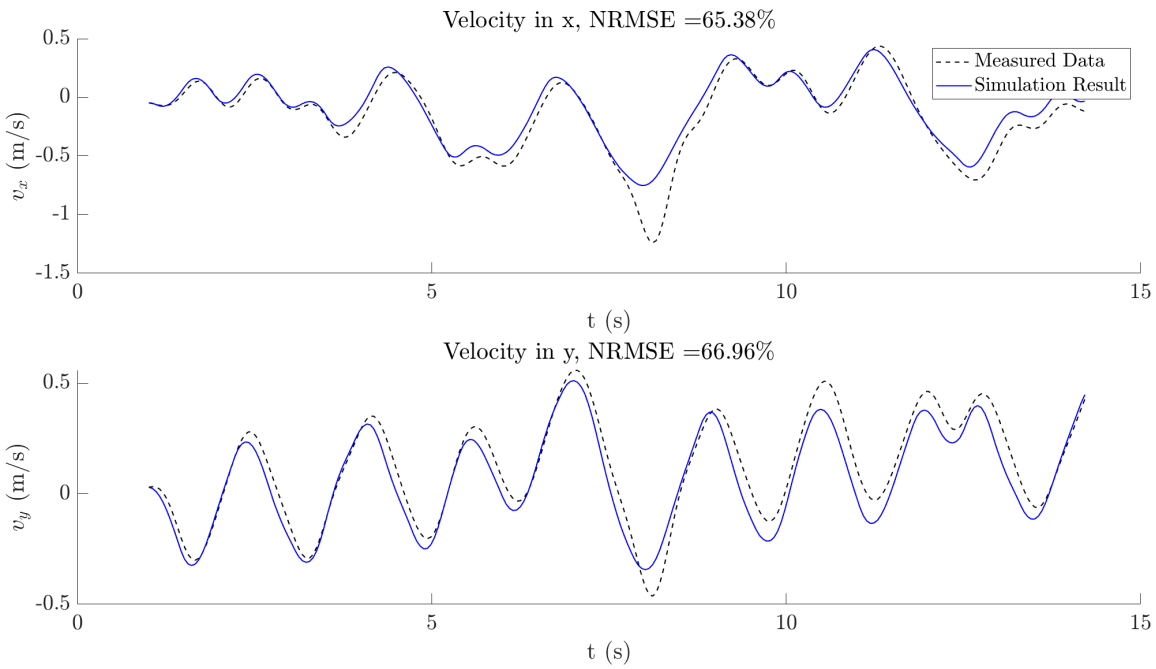

Fig. 33: Simulation results of velocity dynamics for 4^{th} order model over best fitting data set

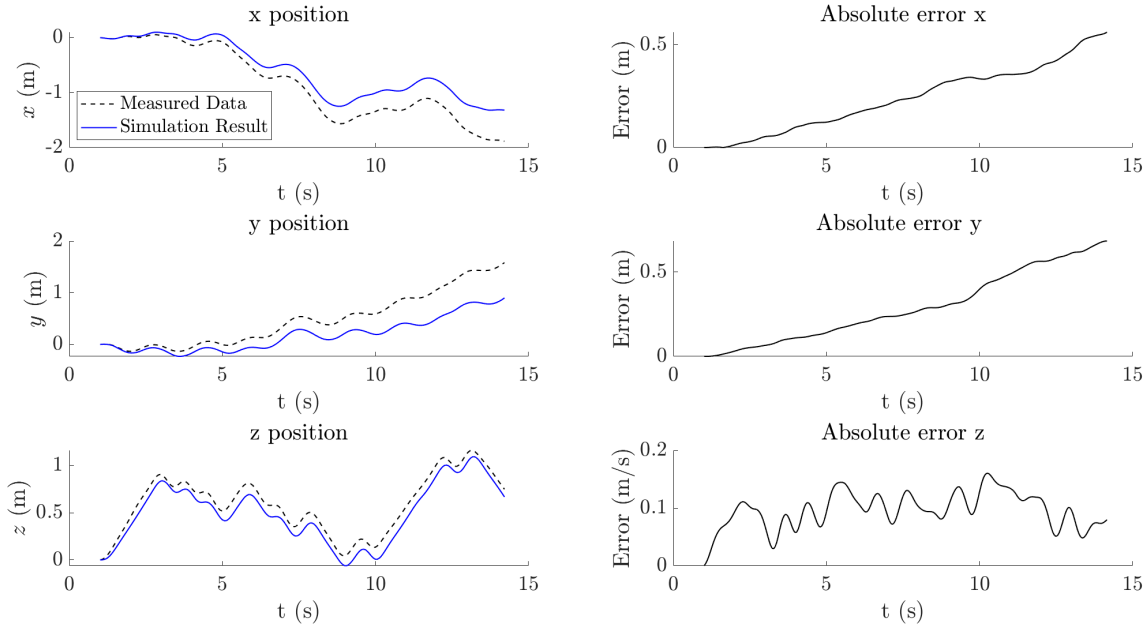

Fig. 34: Simulation results of position dynamics for 4^{th} order model over best fitting data set

APPENDIX M
THRUST SATURATION

During flight it may occur that a combination of input commands requires a total thrust magnitude that is larger than can be delivered with the 4 available rotors. The maximum thrust that can be generated is physically limited by the specifications of the rotors.

If this occurs when the drone is given a single input, such as a commanded vertical velocity, a saturation of the thrust may result in a lower vertical velocity than commanded. When the drone is given multiple inputs, such as a pitch angle and a desired vertical velocity, the internal control loop may choose to satisfy one input over the other.

Identifying this possible behaviour is not in the scope of this research, but the thrust during experiments was analyzed to see if a maximum was reached during flight patterns used for system identification. The thrust is given by the following equation.

$$T = \frac{m(g + \dot{v}_z)}{\cos(\phi)\cos(\theta)}$$

In figure 35 it can be seen that the thrust does not reach a constant maximum value.

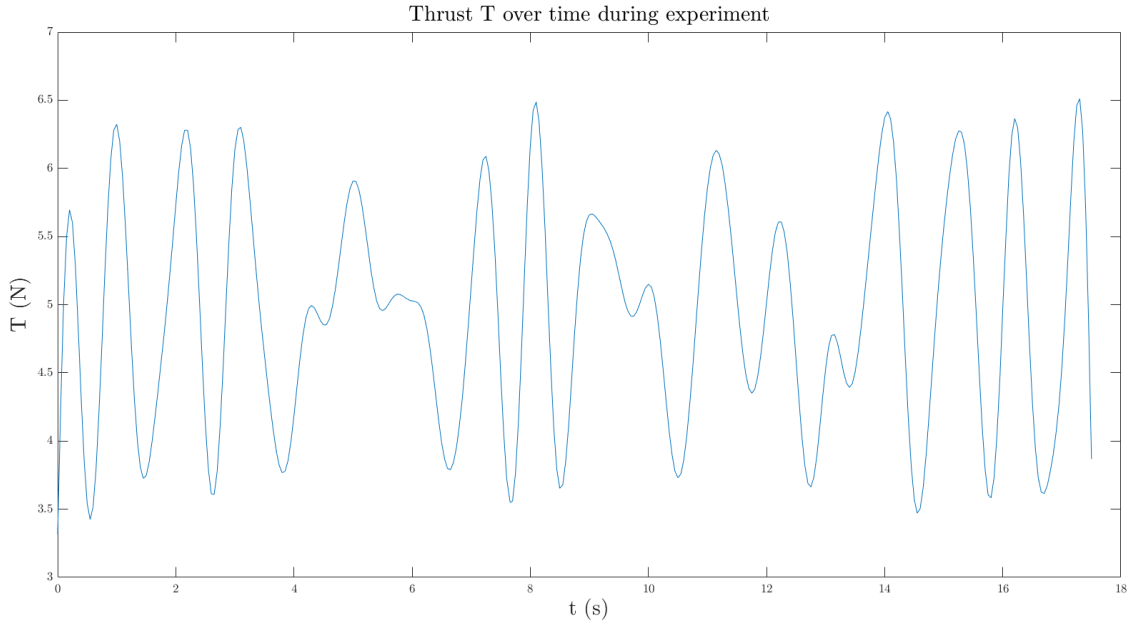

Fig. 35: Thrust level during experiment with concurrent sinusoidal inputs for roll, pitch and vertical velocity of amplitude 12 degrees and 1 $\frac{m}{s}$. It can be seen that the magnitude of the thrust does not reach a steady maximum value.