Modeling and Identification of Multirotor Drone Dynamics for Onboard MPC Motion Planning

Mathias Bos[†*], Bart Theys[‡], Jan Swevers[†] and Goele Pipeleers[†]
† MECO Research Team, Department of Mechanical Engineering, KU Leuven,
DMMS lab, Flanders Make, Leuven, Belgium
‡ Robotics Research Group, Department of Mechanical Engineering, KU Leuven,
ROB lab, Flanders Make, Leuven, Belgium

ABSTRACT

This paper presents a nonlinear model of multirotor drone dynamics selected specifically to be suited for onboard motion planning using Model Predictive Control (MPC), and presents a generally applicable procedure to identify the model parameters solely based on outdoor flight data. The model stems from a trade-off between prediction accuracy and computational complexity, approximating attitude and thrust control loops as low order linear time-invariant subsystems but without linearizing the thrust vector orientation, and including linear aerodynamic drag and a simplified battery voltage dependency. An open loop simulation compared to recorded data from a free flight maneuver motivates the proposed model complexity in contrast to further simplifications of the proposed model.

1 INTRODUCTION

Despite the current state of computing hardware with a form factor that is convenient for compact drones, onboard real-time motion planning and control remains a challenge. Fully autonomous operation requires state estimation, motion planning and control all to be executed onboard at a reasonable update rate. One of the promising techniques for advanced motion planning and control is Model Predictive Control (MPC). Literature shows this technique already being applied for drone control and navigation, most often restricted to position reference tracking over short horizons using linearized dynamics to limit the computational load [1].
The modeling for simultaneous motion planning and control using MPC asks for a trade-off between sufficient model prediction accuracy and limited model complexity, to reduce the computational load of solving a finite-horizon optimal control problem. In literature, quadrotor models with varying complexity and accuracy have been established, a short review of which now follows.

*Email address: mathias.bos@kuleuven.be
ORCID: 0000-0002-5471-6691

1.1 Related work

A detailed survey of existing kinematic and dynamic models of quadrotors and their derivation is presented in [2]. This survey covers how in general these models can be derived using the Newton-Euler method or the Euler-Lagrange formalism, and how given some assumptions on structure rigidity, symmetry and center of gravity, the most commonly used basic quadrotor model is derived starting from Newton's second law. This basic nonlinear model can be formulated as

$$\begin{cases} \dot{\boldsymbol{p}} &= \boldsymbol{v} \\ \dot{\boldsymbol{v}} &= \boldsymbol{g} + \frac{1}{m}\mathbf{R}\boldsymbol{f}_t \\ \mathbf{J}\dot{\boldsymbol{\omega}} &= \boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega}, \end{cases} \tag{1}$$

where $\boldsymbol{p}$ is the drone position vector, $\boldsymbol{v}$ the drone velocity vector, $\boldsymbol{g}$ the gravitational acceleration vector, m the drone mass, $\mathbf{R}$ the rotation matrix from the body to the world frame, $\boldsymbol{f}_t$ the total thrust force, $\boldsymbol{\tau}$ the body torques, $\boldsymbol{\omega}$ the angular rate vector and $\mathbf{J}$ the drone inertia matrix. The thrust force equals the vector sum of all motor thrusts. Often they are assumed to be aligned with the vertical body axis, such that the thrust magnitude equals the sum of the individual motor thrusts. Body torques can be expressed as a function of the thrust forces and the quadrotor geometry.

The set of equations in Equation 1 still misses a relation between the orientation, here represented by the rotation matrix $\mathbf{R}$, and the angular rate vector $\boldsymbol{\omega} = [p, q, r]^\top$. This relation is usually constituted in one of three ways, depending on the used representation of the drone body orientation: through roll-pitch-yaw Euler angles and their derivatives, through quaternion derivatives, or through the derivative of the rotation matrix itself. In terms of the roll-pitch-yaw Euler angles ϕ, θ and ψ, $\mathbf{R}$ is, introducing $c_\gamma \triangleq cos(\gamma)$, $s_\gamma \triangleq sin(\gamma)$, $t_\gamma \triangleq tan(\gamma)$:

$$\begin{aligned} \mathbf{R} &= \mathbf{R}_\psi \mathbf{R}_\theta \mathbf{R}_\phi \\ &= \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix}. \end{aligned} \tag{2}$$

The Euler angle derivatives are in that case related to the angular rates by

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} \boldsymbol{\omega}. \tag{3}$$

When using the quaternion rotation formulation $\mathbf{R}(\boldsymbol{q})$, the derivative of the orientation uses the Hamilton quaternion multiplication to express the derivative of the quaternion as $\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \odot \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}$ [3]. In the third alternative, the derivative of $\mathbf{R}$ can be formulated directly as $\dot{\mathbf{R}} = \mathbf{R}\lfloor\boldsymbol{\omega}\times\rfloor$, where $\lfloor\boldsymbol{\omega}\times\rfloor$ is the skew-symmetric matrix formed with the elements of $\boldsymbol{\omega}$ [4].

The basic model in Equation 1 does not account for aerodynamic effects, which is often sufficient for applications close to the hovering regime. In [2] and [5], a number of aerodynamic effects and ways to model them are listed, such as the ground and ceiling effect, the effect of the angle of attack with respect to the free stream, blade flapping, and interference caused by the vehicle body in the slip stream of the rotor. Often, an approximation that is linear in the translational velocity is used such as in [6], as this describes the dominant aerodynamic effects fairly well even up to significant velocities. The velocity induced aerodynamic drag force is then given by $\boldsymbol{f}_v = -\mathbf{R}\mathbf{D}\mathbf{R}^\top\boldsymbol{v}$, with $\mathbf{D}$ a diagonal matrix containing drag coefficients. This drag force is added to the second line in Equation 1.

A very common approach to avoid the complexity and nonlinearity of the standard nonlinear model, is to perform a linearization around the hover state and assume small angular deviations from this state. Also for drone MPC this is a popular approach, as it alleviates the computational burden, but doing so sacrifices prediction accuracy for more dynamic behavior that deviates significantly from the hover state [1].

Another approach to approximate the full drone model, is to assume that an attitude controller is already in place, which is either assumed to track attitude references perfectly as in [7], or responds as a torsional inertia-spring-damper SISO system in a fully linear rotational and translational model as in [8].

Lastly it can be noted that more complex and detailed models exist, such as [9], in which brushless DC motors and electronic speed controller models are included. However, to the best of our knowledge these detailed models are not used in (online) motion planning and control.

A range of system identification methods is discussed in [2], covering identification procedures for first principles-based models, also referred to as white box models, for (linearized) grey box models, and data-based black box models. Often, however, model parameters are retrieved in a cumbersome procedure from CAD models, which requires detailed models for all parts of the quadrotor, and test benches for separate components such as the motors and propellers [10].

1.2 Contribution and paper structure

This paper proposes a nonlinear multirotor drone model without a small angle assumption that is specifically selected to address the trade-off between model complexity and computational efficiency, together with a simple procedure to identify the unknown model parameters, that does not require special facilities other than the sensors that are present in a realistic outdoor drone application and that do not require dismantling the drone to identify components separately.

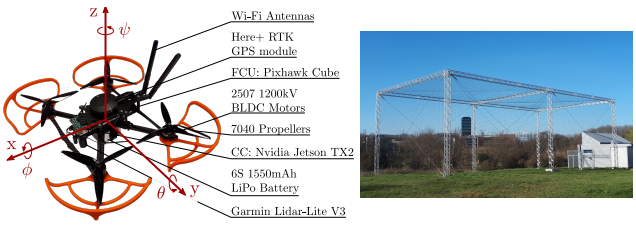

Figure 1: Quadrotor used in the experiments, displaying the axis convention on the body frame with the roll-pitch-yaw Euler angles, and outdoor infrastructure with safety nets.

The paper is structured as follows. First Section 2 presents the model, next Section 3 describes the identification procedure and parameter fitting results, and finally in Section 4 a free flight experiment qualitatively validates the proposed model and identification procedure. This validation compares the simulation of the proposed model to data recorded in the experiment, and compares the simulation results with more simplified versions of the model, justifying the inclusion of nonlinearity, drag and battery voltage dependency. Lastly the limitations to the applicability of the model are discussed.

2 PROPOSED DYNAMIC MODEL

The proposed model simplifies the ensemble of the drone dynamics and the control cascade of attitude control, angular rate control and motor control into a structure that takes throttle input, roll angle, pitch angle and yaw rate references as input, which is the control level often referred to as the stabilize or angle flight mode in commonly used flight controllers. The choice for attitude-throttle inputs in the proposed model is motivated by the applicability on the popular hardware setup where a Companion Computer (CC) performs high-level optimal control and sends reference inputs to the Flight Control Unit (FCU)[1]. This setup shows a significant communication and processing delay in the order of 0.1 s and the update frequency of state estimates from the FCU to the CC is limited at $50 - 100$ Hz, rendering it infeasible to perform attitude control which requires a control rate in the order of 400 Hz [11]. Control with attitude reference inputs however has been demonstrated with input frequencies around 50 Hz in [8] and [12], which is realistic for the given setup and still allows dynamic control.

The proposed model is schematically shown in Figure 2. The schematic reads from the top, starting with the control inputs, to the bottom, towards the output which is the drone

[1]Computation hardware used for this paper:
FCU: Pixhawk 2.1 Cube Black running ArduCopter 3.7-dev firmware. (`https://github.com/unl-nimbus-lab/ardupilot`)
CC: Nvidia Jetson TX2 running Jetpack 4.3.
Serial communication with MAVLink using the Mavros library.

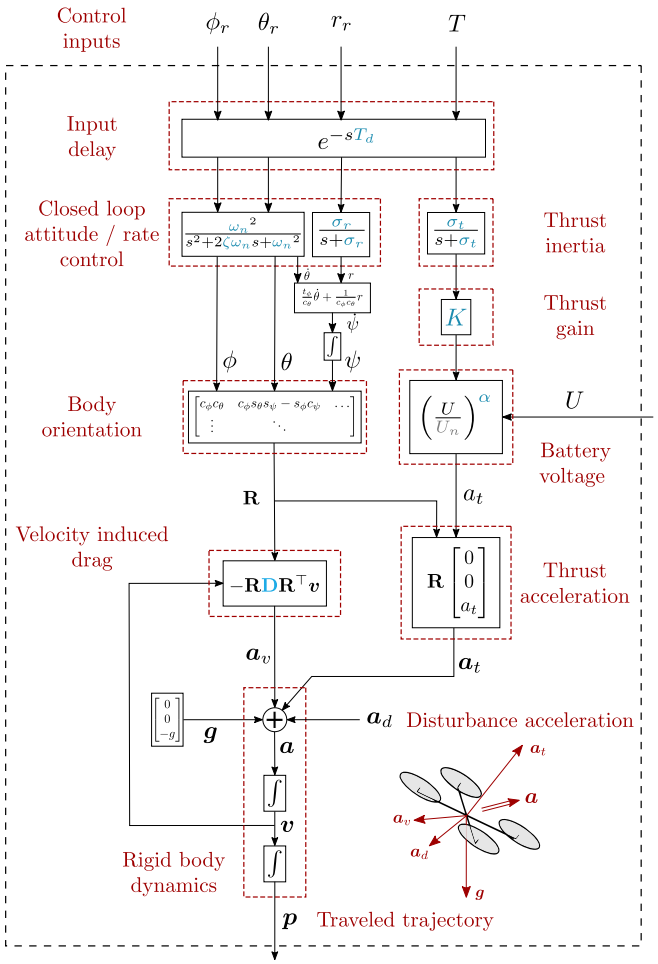

Figure 2: Schematic representation of the proposed model.

position or motion. The control inputs are the roll angle, pitch angle and yaw rate references ϕ_r, θ_r, r_r and the throttle input T. The communication and processing delay between the CC and the FCU is approximated by lumping it into an input delay. The responses of the attitude and thrust with respect to the delayed references are modeled as linear time-invariant second order systems for the roll and pitch, and first order systems for the yaw rate and thrust. The delayed and filtered throttle input is multiplied by a thrust gain and a battery voltage dependent gain, as the battery voltage affects the delivered thrust for a given throttle input. $\mathbf{R}$, the rotation matrix from the body to the world frame, is composed with the Euler angles ϕ, θ and ψ, following the axis convention displayed in Figure 1. It is not linearized, allowing for an accurate representation of large tilt angles. The orientation and the thrust acceleration magnitude a_t are prerequisites to obtain the rigid body dynamics.

The rigid body dynamics are considered as those of a point mass on which four forces act, represented by an equivalent acceleration through division by the drone mass: thrust acceleration $\boldsymbol{a}_t$, velocity induced drag acceleration $\boldsymbol{a}_v$, gravitational acceleration $\boldsymbol{g}$ and an unmodeled disturbance acceleration $\boldsymbol{a}_d$. The thrust acceleration and the drag acceleration closely follow the proposed model from Faessler et al. [6]. The thrust acceleration, corresponding to Faessler's mass-normalized collective thrust term, is assumed to be oriented along the body z-axis. Axial rotor drag causing thrust loss is not accounted for in the thrust acceleration, but instead is included in the drag acceleration. The drag acceleration follows Faessler's 'RDRv' model with $\mathbf{D}$ a constant diagonal matrix containing drag acceleration coefficients. Complex aerodynamic effects as introduced in Section 1 are neglected. The velocity and position are obtained through integration of the acceleration.

The equations of the resulting nonlinear state space model, which is of the form $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\Pi})$, are given in Equation 4. A list of all variables and constants is given in Table 1. The set of twelve parameters $\boldsymbol{\Pi}$ that fully characterize the system are highlighted in blue and are listed in Table 2. How to obtain their values is discussed in the next section. Overbarred symbols represent delayed inputs as in $\bar{u} = u(t - T_d)$. Vectors and their components are expressed in the inertial world frame.

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{p}_z \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \ddot{\phi} \\ \ddot{\theta} \\ \dot{r} \\ \dot{\boldsymbol{a}}_t \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \dot{v}_x^{(*)} \\ \dot{v}_y^{(*)} \\ \dot{v}_z^{(*)} \\ \dot{\phi} \\ \dot{\theta} \\ \frac{t_\phi}{c_\theta}\dot{\theta} + \frac{1}{c_\phi c_\theta} r \\ -{\omega_{n,\phi}}^2 \phi - 2\zeta_\phi \omega_{n,\phi} \dot{\phi} + {\omega_{n,\phi}}^2 \bar{\phi}_r \\ -{\omega_{n,\theta}}^2 \theta - 2\zeta_\theta \omega_{n,\theta} \dot{\theta} + {\omega_{n,\theta}}^2 \bar{\theta}_r \\ -\sigma_r r + \sigma_r \bar{r}_r \\ -\sigma_t \boldsymbol{a}_t + K \left(\frac{U}{U_n}\right)^\alpha \sigma_t \bar{T} \end{bmatrix} \tag{4}$$

$$(*)\ \underbrace{\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix}}_{\dot{\boldsymbol{v}}} = \underbrace{\mathbf{R} \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{a}_t \end{bmatrix}}_{\boldsymbol{a}_t} \underbrace{-\mathbf{RDR}^\top \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}}_{\boldsymbol{a}_v} \underbrace{- \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}}_{\boldsymbol{g}}$$

In this equation the rotation matrix $\mathbf{R}$ from the body to the world frame in terms of the roll-pitch-yaw Euler angles is defined as was described in Section 1.

3 PARAMETER ESTIMATION

To identify the parameters of the selected model, we follow a procedure solely based on flight data, unlike previous works describing cumbersome identification procedures on test benches or based on CAD models. This procedure broadens the applicability towards both open and closed source

Symbol	Variable	Physical units
T	Throttle input	-
ϕ	Roll Euler angle	rad
θ	Pitch Euler angle	rad
ψ	Yaw Euler angle	rad
ϕ_r	Roll angle reference	rad
θ_r	Pitch angle reference	rad
$\dot{\phi}$	Roll Euler rate	rad/s
$\dot{\theta}$	Pitch Euler rate	rad/s
$\dot{\psi}$	Yaw Euler rate	rad/s
p	Roll rate	rad/s
q	Pitch rate	rad/s
r	Yaw rate	rad/s
r_r	Yaw rate reference	rad/s
U	Battery voltage	V
$\mathbf{R}$	Rotation matrix	-
$\boldsymbol{a}$	Acceleration vector	m/s^2
a	Acceleration magnitude	m/s^2
$\boldsymbol{v}$	Translational velocity vector (ground speed)	m/s
$\boldsymbol{p}$	Position vector	m
$\boldsymbol{a}_t$	Thrust acceleration vector	m/s^2
a_t	Thrust acceleration magnitude	m/s^2
$\boldsymbol{a}_v$	Velocity induced drag acceleration vector	m/s^2
$\boldsymbol{a}_d$	Disturbance acceleration vector	m/s^2
$\boldsymbol{g}$	Gravitational acceleration vector	m/s^2
s	Laplace variable	s^{-1}
Symbol	**Constant**	**Value**
g	Gravitational constant	9.81 m/s^2
U_n	Nominal battery voltage	22.2 V

Table 1: Model variables and constants with their symbol and physical units.

Symbol	Parameter	Value
$\omega_{n,\phi}$	Roll natural frequency	25.43 rad/s
$\omega_{n,\theta}$	Pitch natural frequency	23.40 rad/s
ζ_ϕ	Roll damping ratio	0.44
ζ_θ	Pitch damping ratio	0.39
σ_r	Yaw rate decay constant	21.54 s^{-1}
σ_t	Thrust decay constant	20.36 s^{-1}
K	Thrust gain	54.55 m/s^2
α	Battery voltage exponent	1.71
d_x	Drag acceleration coefficient x	0.39 s^{-1}
d_y	Drag acceleration coefficient y	0.39 s^{-1}
d_z	Drag acceleration coefficient z	0.51 s^{-1}
$\mathbf{D}$	Drag acceleration coefficient matrix	$diag([d_x, d_y, d_z])$
T_d	Communication delay time	0.1 s

Table 2: Model parameters with their symbol and identified value.

platforms. Moreover, the procedure only uses sensor information that is available in relevant, outdoor scenarios.

To gather the data to estimate the parameters $\mathbf{\Pi}$, we perform three distinct experiments that each apply a maneuver repeatedly for varying input conditions, as illustrated in Figure 3. Decoupling the dynamics by applying step inputs on a subset of the inputs allows to assess the quality of the resulting fit more easily. The maneuvers can be executed within limited space, i.e. in this work specifically a flight zone of $30 \times 15 \times 10$ m shown in Figure 1. Each experiment results in M time series of varying length N. The sensors used by the FCU are 1) an integrated IMU [2], 2) a Here+ GPS module, 3) a lidar altitude rangefinder (Garmin LIDAR-Lite v3). As a consequence of the approximate modeling of closed loop subsystems, the full procedure including all maneuvers must be repeated as soon as one of the components (e.g. battery, propellers) of the drone setup changes.

Each of the maneuvers applies step inputs of varying magnitude. Firstly, the roll/pitch maneuver applies roll or pitch reference step inputs of 0.2 rad, 0.3 rad and 0.5 rad. The roll and pitch parameters $\mathbf{\Pi}_1 = [\omega_{n,\phi}, \zeta_\phi, \omega_{n,\theta}, \zeta_\theta]^\top$ are identified on the smaller steps of 0.2 rad, 0.3 rad to prevent angular rate saturation. Each of the steps is repeated three times to average measurement noise and random disturbances. Secondly, the yaw rate maneuver applies yaw rate reference step inputs of 1 rad/s and 2 rad/s, each repeated twice, to identify the yaw rate parameter $\mathbf{\Pi}_2 = \sigma_r$. Thirdly, the thrust maneuver applies varying throttle step inputs as illustrated in Figure 3 while recording the vertical acceleration and the battery voltage to gather data for ascending, descending and near to hovering conditions. The throttle input switches from T_0 at time t_0 to T_1 at t_1. Red arrows qualitatively indicate the magnitude of the applied throttle input. The thrust maneuver is executed for 18 combinations of T_0 and T_1, with T_0 ranging from 0.05 to 0.30 and T_1 ranging from 0.10 to 0.40, both in steps of 0.05. Each of the combinations is repeated three to six times over the battery voltage range from 21.5 V up to 24.5 V, resulting in a total of 72 recorded step responses. This serves to identify the thrust parameters $\mathbf{\Pi}_3 = [\sigma_t, K, \alpha]^\top$. The recorded orientation and acceleration in the roll/pitch maneuver allow to estimate the drag parameters $\mathbf{\Pi}_4 = [d_x, d_y, d_z]^\top$ after an initial fit of the thrust parameters. Only the data with roll/pitch reference steps of 0.3 rad and 0.5 rad is retained to assure sufficiently high acceleration such that the unmodeled disturbance acceleration is relatively low compared to the thrust acceleration and drag acceleration. For the same reason it is also important that the roll/pitch maneuvers are performed in near to windless conditions. Because the thrust identification is more accurate given accurate drag parameters and vice versa, their parameter estimates are updated iteratively until convergence.

With the gathered data, first, the delay time T_d is found

[2] LSM303D integrated accelerometer / magnetometer, L3GD20 gyroscope, MPU9250 Gyro / Accel

$\boldsymbol{\Pi}_j$	$e_{i,k}$	$\dot{\boldsymbol{x}} = \boldsymbol{f}_j(\boldsymbol{x}, \boldsymbol{u}, \Pi)$
$\boldsymbol{\Pi}_1$	$\widetilde{\phi}_{i,k} - \phi_{i,k}$	$\begin{bmatrix}\dot{\phi}\\ \ddot{\phi}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -{\omega_{n,\phi}}^2 & -2\zeta_\phi\omega_{n,\phi}\end{bmatrix}\begin{bmatrix}\phi\\ \dot{\phi}\end{bmatrix} + \begin{bmatrix}0\\ {\omega_{n,\phi}}^2\end{bmatrix}\bar{\phi}$
	$\widetilde{\theta}_{i,k} - \theta i,k$	$\begin{bmatrix}\dot{\theta}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -{\omega_{n,\theta}}^2 & -2\zeta_\theta\omega_{n,\theta}\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} + \begin{bmatrix}0\\ {\omega_{n,\theta}}^2\end{bmatrix}\bar{\theta}$
$\boldsymbol{\Pi}_2$	$\widetilde{r}_{i,k} - r_{i,k}$	$\dot{r} = -\sigma_r r + \sigma_r \bar{r}_r$
$\boldsymbol{\Pi}_3$	$\widetilde{a}_{z,i,k} - a_{z,i,k}$	$\begin{bmatrix}\dot{v}_z\\ \dot{a}_t\end{bmatrix} = \begin{bmatrix}a_t - D_z v_z - g\\ -\sigma_t a_t + K\left(\frac{U}{U_n}\right)^{\alpha}\sigma_t\bar{T}\end{bmatrix}$
$\boldsymbol{\Pi}_4$	$\|\widetilde{\boldsymbol{a}}_{i,k} - \boldsymbol{a}_{i,k}\|$	$\begin{bmatrix}\dot{v}_x\\ \dot{v}_y\\ \dot{v}_z\\ \dot{a}_t\end{bmatrix} = \begin{bmatrix}\mathbf{R}\begin{bmatrix}0\\0\\a_t\end{bmatrix} - \mathbf{R}\mathbf{D}\mathbf{R}^\top\boldsymbol{v} + \boldsymbol{g}\\ -\sigma_t a_t + K\left(\frac{U}{U_{nom}}\right)^{\alpha}\sigma_t\bar{T}\end{bmatrix}$

Table 3: Error term and evaluated dynamics for the estimation of each of the parameter subsets. The tilde indicates recorded data.

as the mean of the delay time between applied input and observed change on the corresponding output over all experiment data. Next, each of the remaining parameter subsets $\boldsymbol{\Pi}_j$ is estimated in a least squares optimization:

$$\begin{aligned}\boldsymbol{\Pi}_j &= \underset{\boldsymbol{\Pi}_j}{\operatorname{argmin}}\, V\\ \text{subject to: } \boldsymbol{x}_{i,k+1} &= \boldsymbol{F}_j(\boldsymbol{x}_{i,k}, \bar{\boldsymbol{u}}_{i,k}, \boldsymbol{\Pi}_j) \quad \text{for } k{=}0,1,\ldots,N_i{-}1,\\ &\qquad\qquad\qquad\qquad\quad \text{for } i{=}1,2,\ldots,M\\ \boldsymbol{x}_{i,0} &= \widetilde{\boldsymbol{x}}_{i,0}, \qquad\qquad\qquad \text{for } i{=}1,2,\ldots,M\end{aligned} \tag{5}$$

where

$$V = \sum_{i=1}^{M} V_i = \sum_{i=1}^{M}\sum_{k=0}^{N_i-1} e_{i,k}^2$$

$$\boldsymbol{F}_j(\boldsymbol{x}_k, \bar{\boldsymbol{u}}_k, \boldsymbol{\Pi}_j) = \int_{t_k}^{t_{k+1}} \boldsymbol{f}_j(\boldsymbol{x}_k, \bar{\boldsymbol{u}}_k, \boldsymbol{\Pi}_j)dt. \tag{6}$$

The integral from Equation 6 is evaluated using a fourth order Runge Kutta integration scheme. The error terms $e_{i,k}$ and the evaluated dynamics $\boldsymbol{f}_j$ for each of the identified parameter subsets are given in Table 3.

This least squares problem is solved for each of the parameter subsets using CasADi [13]. The resulting estimates for the parameters are given by Table 2. Figure 4 shows the measured roll, yaw rate and throttle input step responses together with the simulated results using the obtained parameters. The result for the pitch step responses is fully analogous to the roll step responses and is therefore not shown. The roll, yaw rate and throttle input step response data, on the top left, top right, and in the middle respectively, clearly show delayed predominantly second order and first order behavior, captured well by the model simulation. The middle plot shows the vertical acceleration as a function of time starting from t_1 and as a function of battery voltage for fifteen repetitions of the thrust maneuver, showing only throttle input combinations $T_0 = 0.05,\ 0.10,\ 0.15$ and $T_1 = 0.35$ for visual clarity. The acceleration magnitude, shown as a function

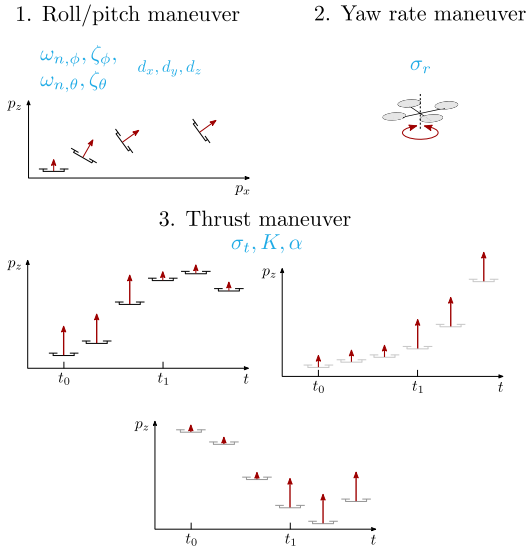

Figure 3: Illustration of identification experiment maneuvers, along with the parameters identified in the experiment.

of both time and velocity in the bottom plots, is predicted remarkably better with the inclusion of linear drag compared to the simulation without any drag. The arrows indicate the magnitude of the estimated drag acceleration.

4 RESULTS AND DISCUSSION

To validate the prediction quality of the model and to motivate the inclusion of the modeled effects, its simulation on inputs from the first 1.5 s of a free flight using a rudimentary MPC controller is compared with the simulation of simplified versions of the model and the recorded experimental data. During this experiment, the battery voltage is around 23 V.

The models under comparison are 1) the full model as in Figure 2 and Equation 4, 2) a linearization of this model, treating yaw as described in [4], 3) the proposed model excluding drag, 4) the proposed model excluding the battery voltage dependency by assuming nominal battery voltage throughout the simulation. The roll, yaw rate and yaw responses are also compared to the situation where no first order or second order behavior is included and only the communication delay is considered. The adaptations to the full model to obtain the simplified versions are given by Table 4.

The simulation results in Figure 5, in which all plots correspond to the same short maneuver, prove that the full model predicts the drone motion significantly more accurately than the simplified versions of the model. Compared to the recorded data and the full model simulation, the only delay approximation clearly misses distinct evolutions of the attitude responses because it omits the first and second order behavior, as seen in the p, ϕ, r and ψ plots. The linear approximation strongly overestimates the vertical thrust component because of the assumption of small angles with respect to the

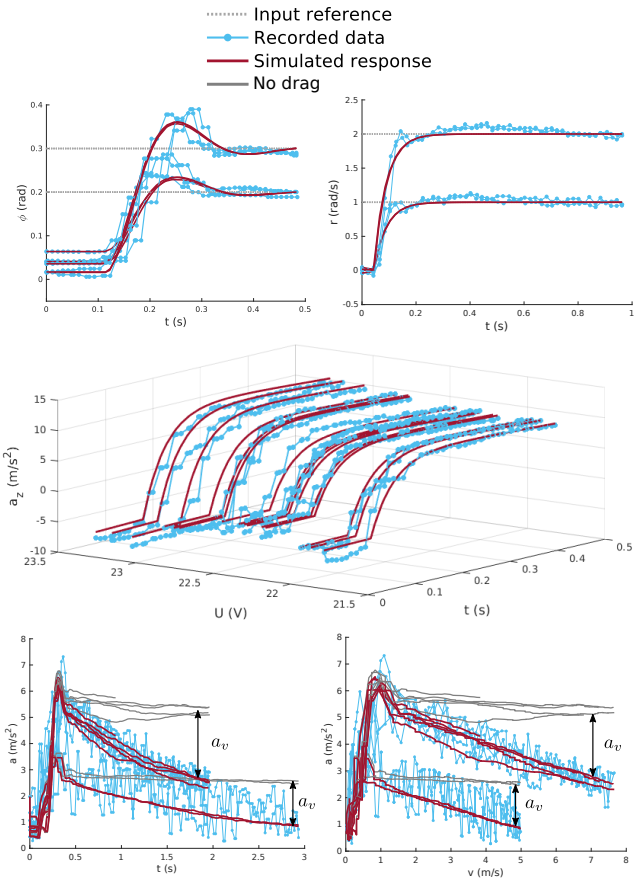

Figure 4: Validation of the model with estimated parameters.

Model variation	**Model / change to the full model**
Full model	$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\Pi})$
Linearized	$\delta\dot{\boldsymbol{x}} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right\|_{\boldsymbol{x}^*,\boldsymbol{u}^*} \boldsymbol{x} + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right\|_{\boldsymbol{x}^*,\boldsymbol{u}^*} \boldsymbol{u}$, $\boldsymbol{x} = \boldsymbol{x}^* + \delta\boldsymbol{x}$, with $\boldsymbol{x}^*$ and $\boldsymbol{u}^*$ the hover state and hover inputs
No drag	$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\Pi})$ with $d_x = d_y = d_z = 0$
No battery dependency	$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\Pi})$ with $\alpha = 0$
Only delay	$\phi = \bar{\phi}_r$, $\theta = \bar{\theta}_r$, $r = \bar{r}_r$

Table 4: Variations of the full model for the comparison of prediction accuracy.

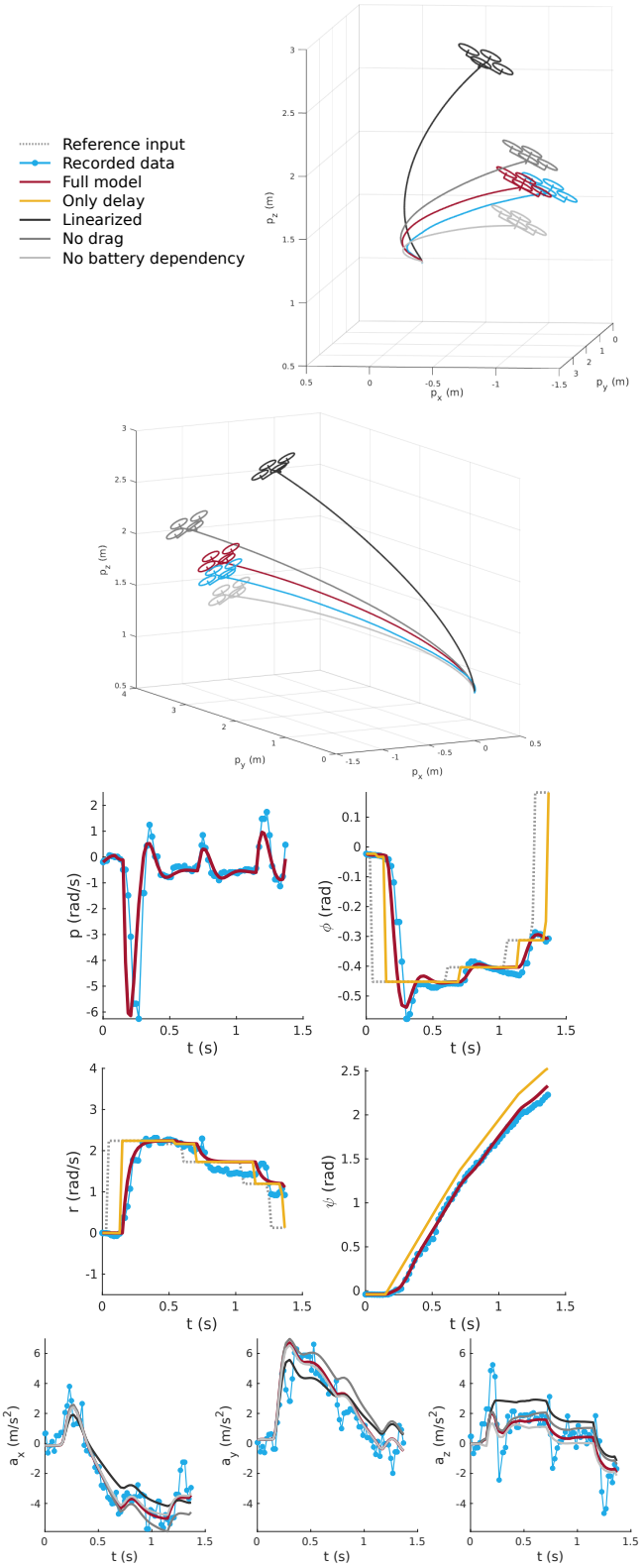

Figure 5: Comparison of four model variations of varying complexity with data recorded in the validation experiment. From top to bottom: 1) drone position, 2) roll rate and roll, yaw rate and yaw, 3) acceleration components in the world frame. All these plots correspond to the same short maneuver.

hover state, which is seen in the a_z and 3D position plots. The no drag approximation neglects the velocity induced counteracting force, hence overestimating the achieved acceleration, most clearly visible in the a_y and 3D position plots. The no battery dependency approximation neglects the effect of the higher battery voltage of 23 V compared to the nominal battery voltage of 22.2 V, hence underestimating the generated thrust and therefore acceleration. Consequently, the 3D position plots show a lower altitude for this simulation. These observations confirm the choice to include the low order approximation of low-level control loops, nonlinearity, drag effect and battery voltage dependency in the model.

To conclude this discussion, we discuss the limitations to the applicability of the proposed model that the approximations of the done dynamics made in this paper bring. Firstly, the low order decoupled approximation of control loops using an Euler angle representation and the assumption of linear drag restrict the use to low or medium velocity applications with limited roll and pitch angles. Secondly, saturation on the angular rates is not taken into account, meaning that for accurate predictions the high level planner should not request subsequent angle references that exceed the rate limits. Thirdly, simulating with attitude and throttle as inputs while neglecting a possible accelerometer bias in the identification procedure entails large drift on the velocity and position predictions over longer horizons, as the prediction error on the orientation and acceleration is integrated twice over time to obtain the position. This high level of uncertainty demands frequent corrections when using the model for estimation and control, which is not a problem in the intended MPC setting as this control strategy inherently comprises state feedback.

5 CONCLUSION AND FUTURE WORK

This paper presented a multirotor drone model and a procedure to identify its unknown model parameters using the data measured during three simple maneuvers that do not require special facilities other than the sensors that are present in a realistic outdoor drone application and that do not require dismantling the drone to identify components separately. The model was crafted to address the trade-off between sufficient prediction accuracy and limited model complexity that arises when considering autonomous drone applications. The complexity of the representation of the low level control systems steering the attitude and thrust was reduced by modeling them as low order linear time-invariant subsystems. Linear drag and simplified battery voltage dependency were proven to benefit the prediction accuracy when included in the simulation.

In future work we will exploit this model in MPC, aiming for outdoor applications with a requirement for dynamic autonomous control. Early tests show approximate planning update rates of 20 Hz and more for time horizons over 10 s, which, possibly supported by (linear) feedback control, is adequate for many applications. Another interesting use of this model is the estimation of force disturbances, which were left untouched in this paper. Disturbance estimation could serve to estimate and reject wind influence for improved position tracking performance, and to estimate and compensate for the relative position and swinging of suspended payloads.

ACKNOWLEDGEMENTS

This work has been carried out within the framework of projects Flanders Make SBO MULTISYSLECO: Multi-System Learning Control, Flanders Make SBO MULTIROB: Rigorous approach for programming and optimal control of multirobot systems, and Flanders Make SBO HySLAM: A Hybrid SLAM approach for autonomous mobile systems. Flanders Make is the Flemish strategic research centre for the manufacturing industry. This work also benefits from project G0C4515N of the Research Foundation - Flanders (FWO - Flanders).

REFERENCES

[1] H. Nguyen, M.S. Kamel, K. Alexis, and R. Siegwart. Model predictive control for micro aerial vehicles: A survey. arXiv 2011.11104, 11 2020.

[2] Xiaodong Zhang, Xiaoli Li, Kang Wang, and Yanjun Lu. A survey of modelling and identification of quadrotor robot. *Abstract and Applied Analysis*, 2014:320526, Oct 2014.

[3] Emil Fresk and George Nikolakopoulos. Full quaternion based attitude control for a quadrotor. In *2013 European Control Conference (ECC)*, pages 3864–3869, 2013.

[4] M. Kamel, M. Burri, and R. Siegwart. Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles. arXiv 1611.09240, 2017.

[5] Gabriel Hoffmann, Haomiao Huang, Steven Waslander, and Claire Tomlin. *Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment*.

[6] M. Faessler, A. Franchi, and D. Scaramuzza. Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robotics and Automation Letters*, 3(2):620–626, 2018.

[7] S Li, C De Wagter, CC de Visser, QP Chu, and GCHE de Croon. In-flight model parameter and state estimation using gradient descent for high-speed flight. *International Journal of Micro Air Vehicles*, 11, 2019.

[8] P. Bouffard, A. Aswani, and C. Tomlin. Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *2012 IEEE International Conference on Robotics and Automation*, pages 279–284, 2012.

[9] Bart Theys and Joris De Schutter. Virtual motor torque sensing for multirotor propulsion systems. *IEEE Robotics and Automation Letters*, 6(2):4149–4155, 2021.

[10] A. Chovancová, T. Fico, Ľ. Chovanec, and P. Hubinsk. Mathematical modelling and parameter identification of quadrotor (a survey). *Procedia Engineering*, 96:172 – 181, 2014. Modelling of Mechanical and Mechatronic Systems.

[11] ArduPilot Development Team. Ardupilot project. `github.com/ArduPilot/ardupilot`.

[12] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza. Alphapilot: Autonomous drone racing. arXiv 2005.12813, 2020.

[13] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, Mar 2019.