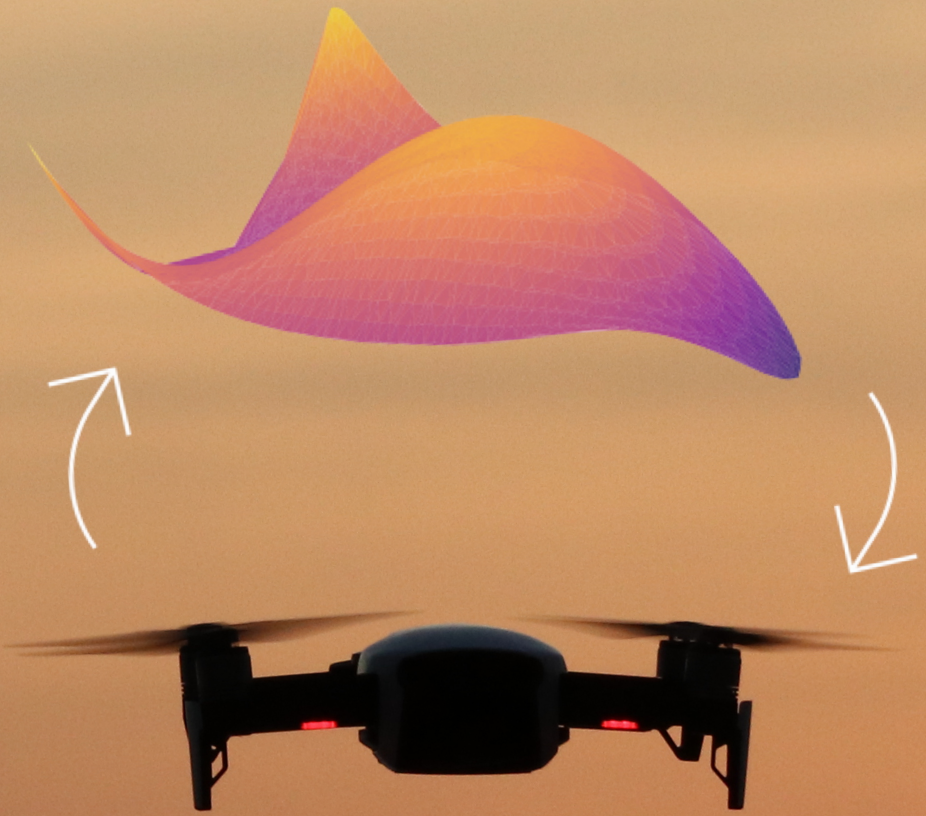
Development of a system identification routine, with prediction reliability estimates, suitable for modelling high-speed and aggressive quadrotor flight outdoors

Jasper J. van Beers



Development of a system identification routine, with prediction reliability estimates, suitable for modelling high-speed and aggressive quadrotor flight outdoors

Using only on-board sensors

to obtain the degree of Master of Science
at the Delft University of Technology,
defended publicly on Tuesday December 21, 2021 at 9:30 AM.

by

Jasper Johannes van Beers

Student number:	4479068	
Project duration:	12 November 2020 - 21 December 2021	
Thesis committee:	Prof. dr. ir. G. C. H. E. de Croon,	TU Delft, Control & Operations
	Dr. ir. C. C. de Visser,	TU Delft, Control & Operations
	Dr. Steven Hulshoff,	TU Delft, Aerodynamics

Delft
University of
Technology

Acknowledgements

Reflecting on my time as a Master's student at TU Delft has made me realize how fortunate I am to be doing the things I enjoy surrounded by supportive peers and caring friends and family. Prior to starting my thesis journey, I was not particularly familiar or interested in quadrotors. However, after working on these vehicles, I can confidently say that I have found a new hobby and research interest. Indeed, the work I have conducted during my thesis, from the programming to the flight tests, has only furthered my interest in research and solidified my desire for a career in this field. However, none of this would have been possible without the guidance of my supervisor, Dr. ir. Coen de Visser. His motivation and drive for advancing knowledge and safety on quadrotors have inspired me to also pursue such endeavours. Thank you for helping me clarify my career interests!

I would also like to thank my girlfriend and close friends who helped me through the tough times of the year and were always supportive. During times of stress, I could always count on these people to help me relax. The meals, laughs, and overall enjoyable times we had over the last year are moments that I truly treasure. Thank you all for your help, kindness and support.

Table of Contents

List of Figures

List of Tables

Part I

Scientific Paper: On prediction intervals

Numerical validation of prediction intervals for artificial neural network quadrotor models

J.J. van Beers [*] and Dr. ir. C.C. de Visser [†]
Delft University of Technology, 2629 HS Delft, The Netherlands

Ensuring the reliability and validity of data-driven quadrotor model predictions is essential for their accepted and practical use. This is especially true for black-box models, such as artificial neural networks (ANNs), for which the mapping of inputs to predictions is ambiguous and subsequent reliability notoriously difficult to ascertain. As prediction intervals (PIs) provide insight into the confidence, and thus reliability, of the models' predictions, two ANN PI estimation techniques - the bootstrap method and the quality-driven direct PI estimation method - are compared with equivalent polynomial PIs and validated numerically for quadrotor models through an existing high-fidelity quadrotor simulation for the first time. It is shown here that the bootstrap method generally mirrors the polynomial PI performance and is robust against noise in the inputs, sporting numerically valid PIs regardless of the noise power. However, it relies heavily on the assumption of normally distributed prediction errors and fails to produce valid PIs when this assumption is not satisfied. Conversely, the quality-driven direct PI estimation method produces valid PIs that are invariant of this assumption but are instead sensitive to high-noise in the measurements insofar as they sometimes fail to enclose the target predictions. Nonetheless, future predictions made by the quality-driven ANN always consistently lie within the interval bounds, regardless of noise, exhibiting a containment of model uncertainty. These results show promise for the use of the evaluated ANN PI estimation methods and imposes some requirements on the identification data, dependent on the chosen estimation technique, to ensure valid estimations.

Contents

[*]Student, Faculty of Aerospace Engineering: Control and Simulation
[†]Assistant professor, Faculty of Aerospace Engineering: Control and Simulation

I. Introduction

Data-driven system identification techniques are often employed to develop models of highly non-linear systems, such as the quadrotor. These data-driven approaches to quadrotor model identification are convenient and attractive due to limited knowledge on analytical descriptions of the quadrotor [1–4]. Subsequently, many quadrotor models have been successfully identified in literature using a plethora of system identification techniques [1, 5–8].

Among these, artificial neural networks (ANNs) are an increasingly popular choice by virtue of their generalizability [6] and aptitude for capturing unknown non-linearities [9, 10]. Consequently, they are employed to capture dynamics in unexplored and extreme regions of the flight envelope. Take, for instance, the use of ANNs to facilitate high-speed (i.e. up to 18 ms^{-1}) quadrotor model identification by Bauersfeld et al. [8]. As many of the current state-of-the-art quadrotor models are identified indoors in constrained and controlled environments (e.g. [1, 8]), ANNs offer an attractive technique to apply to outdoor quadrotor identification where more uncertainties are present and a larger, unknown, region of the flight envelope may be explored.

However, as a black-box modelling technique, it is ambiguous as to how exactly the ANN inputs interact to produce the model outputs. As such, evaluating the reliability and validity of the resultant ANN models is challenging and proves to be a significant barrier to their widespread use in the aerospace industry [11]. Although many ANN quadrotor models have been identified in literature with desirable performance (see, for example, [6–8]), none specify how reliable or valid these models are beyond assessing their performance with respect to some, typically unseen (i.e. validation), flight data sets. Subsequently, an ANN model may produce accurate predictions for both the training and validation data sets and thus may be considered a good model. However, such performance may be constrained to those data sets alone, and may deteriorate due to uncertainties in the inputs which ultimately culminate in erroneous predictions [12]. This is undesirable as it may lead to unpredictable model outputs for similar (e.g. noisy) inputs. Ensuring safety and reliability are paramount to the accepted use of autonomous quadrotors and thus this lack of model reliability measures for ANNs needs to be addressed.

One way to describe the reliability of a model's prediction is through its confidence in that prediction. To this end, the associated prediction intervals (PIs) - the interval wherein a future observation will lie, with a given probability (i.e. confidence) - may be used as a proxy for reliability. For some system identification approaches, such as polynomial regression, there are valid analytical formulations for obtaining these PIs. However, due to the inherent obscurity behind black-box models, analytical formulations that maintain the validity of any estimated PIs are challenging, if not impossible, to derive. This is indeed the case for ANNs.

Consequently, some ANN literature [12–15] propose various methods to estimate these prediction intervals, especially in the face of input uncertainties. One of the more straightforward and convenient approaches is known as the bootstrap method. In this approach, an additional ANN is employed to estimate the PIs associated with the aggregate prediction of an existing ensemble of ANNs [12]. Consequently, it may be applied to an already trained ensemble of ANNs without needing to retrain or augment the underlying model. In their review of popular ANN PI estimation techniques, Khosravi et al. [12] found that the bootstrap approach did well to reflect uncertainty in the inputs. However, the bootstrap method often produces excessively wide PIs, which is undesirable. Alternatively, the prediction intervals may be estimated directly by the (ensemble of) ANNs primarily through an augmentation of the training cost function. Accordingly, Pearce et al. [14] propose a quality-driven direct estimation of the PIs. This approach necessitates a custom cost function - built from PI quality metrics - and training regime for the ANNs which is argued to produce high quality, and valid, PIs [14]. These ANN PI estimation methods, or indeed any others, provide a clear utility in establishing model reliability but have yet to be applied to quadrotor ANN models.

Therefore, due to the promising results of both the bootstrap and quality-driven direct estimation methods, they are applied to quadrotor identification and PI estimation in this paper. However, PIs are only useful if they truly reflect the specified confidence behind making predictions. Consequently, the validity of the estimated PIs from these two methods are evaluated numerically for the first time using noisy data derived from an existing high-fidelity quadrotor simulation platform developed by Sun et al. [16]. This simulation is used to generate mock flight data under controlled conditions for two numerical validation experiments. While not as definitive as an analytical derivation, numerically corroborated PIs nonetheless lend support to their reliability and validity. Moreover, the polynomial prediction intervals, which are known to be reliable given their analytical foundations, are also identified for equivalent polynomial models to act as a benchmark during this numerical validation.

To facilitate this, a brief introduction to the PI quality metrics and estimation methods for the polynomial, bootstrap, and direct methods are given in section II. Subsequently, the procedures for the two numerical validation experiments, including a brief description of the simulation set-up, are provided in section III. The numerical validation results themselves are then summarized in section IV.

II. Model reliability assessment through prediction intervals

Although regression aims to approximate a certain target variable given a set of predictors, measurements of these targets are, in practical applications, often contaminated with bias and noise. This induces a variation in model predictions.

For simplicity, it is commonly assumed that the bias is absent* and the measured targets, $\mathbf{y_m}$, are only influenced by some zero-mean error, $\boldsymbol{\epsilon}$, as given by eq. (1). Here, $\mathbf{y_t}$ denotes the true value of the target [12]. It is often further assumed that the errors, $\boldsymbol{\epsilon}$, are independent and identically distributed [12].

$$\mathbf{y_m} = \mathbf{y_t} + \boldsymbol{\epsilon} \tag{1}$$

Regression models only estimate the measured target which, itself, harbours uncertainties arising from modelling misspecification and the estimation of model parameters. This uncertainty is typically captured through the confidence interval (CI), defined for some confidence level $1 - \alpha$, and is constructed from the variance of the modelling errors. α denotes the alpha level and is typically selected as $\alpha = 0.05$, resulting in a 95% confidence interval (i.e. $1 - \alpha = 0.95$).

However, the uncertainty associated with making predictions encompasses both noise and modelling errors ($\boldsymbol{\sigma}^2_{\hat{\boldsymbol{\epsilon}}}$ and $\boldsymbol{\sigma}^2_{\hat{\mathbf{y}}}$ respectively) as shown in eq. (2). Here, $\boldsymbol{\sigma}^2$ gives the variance associated with the total model predictions and may be used to construct the prediction intervals (PIs) based on a given confidence level $1-\alpha$. The PIs, by definition, encompass the CIs since they consider the errors arising from the measurements of the targets in addition to the modelling errors. In words, the PIs may be interpreted as the "interval in which a future observation will lie, provided that it has previously been observed, up to a specified confidence level". Therefore, the extent of interval may be used to assess the reliability of the prediction wherein narrow intervals denote a reliable prediction due to minimal variation.

$$\boldsymbol{\sigma}^2 = \boldsymbol{\sigma}^2_{\hat{\mathbf{y}}} + \boldsymbol{\sigma}^2_{\hat{\boldsymbol{\epsilon}}} \tag{2}$$

The PI bounds, for a given target value, y_i, and associated variance, σ_i, can be obtained through eq. (3). In eq. (3), $t^{1-\alpha/2}_{N-2}$ gives two sided test statistic for the chosen confidence level, $1 - \alpha$, and n gives the number of observations. Typically, $n = 1$ when making (single) predictions.

$$\hat{y}_i \pm t^{1-\alpha/2}_{N-2} \frac{\sigma_i}{\sqrt{n}} \tag{3}$$

The quality of the PIs are usually assessed through two metrics: the Coverage Probability (PICP) and the Mean Prediction Interval Width (MPIW) [12, 14]. The PICP evaluates the proportion of the (test or validation) target data that is contained within the estimated PIs and is defined by eq. (4). In eq. (4), N denotes the number of test data points, and c_i describes if the target value is contained within the defined bounds of the PI (where L_i = lower bound, U_i = upper bound) for a given confidence level. Therefore, the PICP may be used to verify that the desired confidence level is achieved. For example, PIs designed to capture the target value 95% of the time are expected to maintain a $PICP \geq 95\%$. While the PICP is useful for evaluating the validity of the PIs, it does not itself describe how practical the PIs are since infinitely large PIs also satisfy the PICP requirement.

$$\begin{gathered} PICP = \tfrac{1}{N} \textstyle\sum_{i=1}^{N} c_i \\ \text{where} \\ c_i = \begin{cases} 1, & y_i \in [L_i, U_i] \\ 0, & \text{otherwise} \end{cases} \end{gathered} \tag{4}$$

Complimentary to the PICP, the MPIW is often used to parameterize the width of the PIs. As the absolute widths of the MPIW vary depending on the data set, the MPIW is typically normalized with respect to the range of the target data. The normalized MPIW is defined in eq. (5). Both the PICP and the MPIW should be used in tandem to describe the overall PI quality, seeing as the deficiencies of one metric are perfectly accommodated by the other. Therefore, a 'high-quality' PI is one which satisfies the PICP requirement while being as narrow as possible in terms of the (normalized) MPIW.

$$MIPW = \frac{1}{N} \sum_{i=1}^{N} (U_i - L_i) \tag{5}$$

*For example, eliminated during pre-processing of the data.

These prediction intervals provide a clear utility for interpreting model predictions and may be used to evaluate the subsequent reliability of a given prediction through the validity and width of the intervals.

A. Prediction Intervals for Polynomial models

Conveniently, for a model identified through Ordinary Least Squares (e.g. poylnomial models identified through step-wise regression [1]), $\boldsymbol{\sigma}^2$ may calculated through eq. (6) under the assumptions that the measurement error term, ϵ, is zero-mean, independent and identically distributed. In eq. (6), $\mathbf{x_0}$ denotes the input data for which predictions should be made (i.e. to predict y_0), $\mathbf{X}$ represents the regressor matrix used for training the model, $\mathbf{I}$ gives the identity matrix and σ_e^2 may be approximated through eq. (7).

$$\hat{\sigma}_0^2 = \sigma_e^2 \left(\mathbf{I} + \mathbf{x}_0 \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{x}_0^T \right) \tag{6}$$

$$\hat{\sigma}_e^2 = \frac{1}{N_t - 2} \sum_{i=1}^{N} \hat{e}_i^2 \tag{7}$$

In eq. (7), N_t represents the number of samples used for training and $\hat{e}_i = y_i - \hat{y}_i$ denotes the residual error. Note the difference between ϵ (error due to noise) and e (residual error) used here. The computed $\hat{\sigma}_0$ may then be inputted into eq. (3) to obtain the lower and upper bounds of the PI.

B. Prediction Intervals for ANN models

For ANN models, and indeed many other black box approaches to modelling, obtaining PIs is challenging due to the obscurity of how the model maps the inputs to its predictions. As such, there is no standard technique to obtain these PIs and literature on estimating ANN PIs is varied. Two promising approaches from literature are described in the following sections.

1. Bootstrap method

The bootstrap method is one of the most straightforward and convenient approaches to ANN PI estimation which exploits the inherent stochasticity of the ANN training process. One of the main benefits of the bootstrap method is that it does not require the use of specific network structures or training regimes and can instead be applied to an already trained ensemble of ANNs.

The spread (i.e. variance) of predictions across an ensemble of ANN models, initialized with distinct weights and trained on different subsets of the training data set, gives an approximation of the errors which originate from model uncertainty and misspecification (i.e $\sigma_{\hat{y}}^2$) through the variance across the ensemble predictions [12, 14]. Let the average ensemble prediction for a given target, y_i, be given by eq. (8) where B gives the number of ANN models, and $\hat{y}_i^j$ the prediction of model j.

$$\hat{y}_i = \frac{1}{B} \sum_{j=1}^{B} \hat{y}_i^j \tag{8}$$

Therefore, the variance in model predictions may be constructed through eq. (9) [12, 17].

$$\hat{\sigma}_{y_i}^2 = \frac{1}{B-1} \sum_{j=1}^{B} \left(\hat{y}_i^j - \hat{y}_i \right)^2 \tag{9}$$

The variance due to the measurement errors, σ_{ϵ_i}, is typically unknown and cannot easily be inferred from the ensemble of ANN models. Instead, an additional ANN (hereafter known as the $\sigma_{\epsilon_i}^2$-predictor ANN) may be used to (implicitly) estimate this variance using a different 'unseen' data set (i.e. one that is distinct from that used for training and validating the ANN ensembles [12, 17]).

Using the (trained) ensemble of ANNs, predictions can be made for this unseen data set for which the target values are known and subsequent squared errors taken through eq. (10). Since the variance due to the model uncertainty has been approximated through the variance in ANN predictions in the ensemble (i.e. eq. (9)), the remaining variance in the squared errors, e_i^2, is assumed to be the variance of the measurement noise errors, $\sigma_{\epsilon_i}^2$ [17]. Therefore, a new data set,

$D_r = \{x_i, r_i^2\}_{i=1}^m$, may be constructed linking the inputs, x_i, to the variance residuals, r_i^2, defined in eq. (11). Note the 'max' operator is used here since the variance should be greater than zero.

$$e_i^2 = (y_{m,i} - \hat{y}_i)^2 \tag{10}$$

$$r_i^2 = \max\left(\left(y_{m,i} - \hat{y}_i\right)^2 - \hat{\sigma}_{y_i}^2\right), 0\Big) \tag{11}$$

The $\sigma_{\epsilon_i}^2$-predictor ANN may then be used to implicitly approximate $\sigma_{\epsilon_i}^2$ by minimizing the cost function defined in eq. (12) [12]. This cost function is obtained by taking the natural logarithm of the normal distribution of ϵ and thus ensures that the $\sigma_{\epsilon_i}^2$-predictor ANN produces variances which conform this normal distribution [12]. The practical implementation of this cost function involves setting r_i^2 as the target values (i.e. $r_i^2 \rightarrow y_i$) and $\sigma_{\epsilon_i}^2$ as the ANN predicted values (i.e. $\sigma_{\epsilon_i}^2 \rightarrow \hat{y}_i$). The goal is to minimize this cost function.

$$J = \frac{1}{2}\sum_{i=1}^{m}\left[\ln(\sigma_{\epsilon_i}^2) + \frac{r_i^2}{\sigma_{\epsilon_i}^2}\right] \tag{12}$$

As with the polynomial PIs, eq. (3) may be used to construct the lower and upper PI bounds using the complete estimated model prediction variance, $\hat{\sigma}_i = \hat{\sigma}_{y_i} + \hat{\sigma}_{\epsilon_i}$.

Intuitively, more models used in the ensembles leads to more reliable estimates for the model uncertainty. Consequently, some literature, such as [18, 19], advocate for the use of hundreds to thousands of ANN models. This rather problematic given that training such an extensive collection of models demands significant computational resources and casts doubt on the practical application of the bootstrap method for quadrotor modelling. However, both [12, 14] found that as few as 5-10 ANN models are sufficient for producing reliable prediction intervals. Although, it should be highlighted that the minimum number of ANNs in an ensemble is dependent on the modelling task at hand, and may increase, or decrease, depending on the complexity of employed ANNs and the modelling task itself.

While the bootstrap method may not result in the best quality PIs, often with excessively wide PIs, it is simple to implement and may be applied directly to existing ANNs without interfering with the training process [12]. For quadrotor applications, a more conservative estimate (i.e. wider MPIWs) may be argued to be desirable as it promotes greater caution in predictions. Moreover, the bootstrap method is computationally efficient when making predictions[†] compared to other ANN PI estimation methods and it is found to intuitively reflect uncertainties in the inputs (i.e. higher uncertainties results in wider PIs) [12].

2. *Quality-driven direct estimation*

Instead of estimating the PIs after-the-fact, another approach is to estimate them directly. Although more involved than the bootstrap method, direct access to the training phase allows for the consideration of the PI quality metrics (i.e. $PICP$ and $MPIW$) during training. Such an approach is outlined in the recent work of Pearce et al. [14] wherein an ensemble of FNNs are used to directly predict the upper and lower bounds of the PI associated with a prediction. The PI quality metrics are directly incorporated into the loss function of the ANN training process enabling high-quality PIs to be estimated. Point predictions may then be extracted from the intervals by taking the mean [14]. Consequently, the model will implicitly fit the target data, while optimizing for the PI quality metrics by promoting valid and reliable PIs.

The inspiration behind the approach of Pearce et al. [14] is the foundational work of Khosravi et al. [13] who also previously introduced a PI-based cost function involving the MIPW and PICP in their so called 'Lower Upper Bound Estimation (LUBE)' method. Prior to their contribution, the PI quality metrics were seldom used in tandem. Therefore, Khosravi et al. [13] include both MIPW and PICP in their loss function design and term this function the coverage width-based criterion (CWC). The CWC is defined in eq. (13) where R denotes the range of the target values (i.e. to normalize the MPIW) and λ is a hyper-parameter that modulates the importance of the PICP requirement. The motivation behind this function is to encourage narrow PIs so long as the PICP criterion is met, hence the max operator.

$$CWC = \frac{MPIW}{R}\left(1 + e^{\lambda \max(0, (1-\alpha) - PICP)}\right) \tag{13}$$

[†]Note that this method is, however, computationally demanding during training. This complexity also scales with the number of necessary model ensembles for reliable predictions.

While the LUBE method has since been used successfully in several practical applications (see summary by Pearce et al. [14] and references contained within), Pearce et al. [14] note a few critiques with the original CWC and propose some modifications to improve usability and performance:

- **Multiplicative effect of the MPIW:** Due to the multiplicative effect of the $MPIW$ in eq. (13), there is a global optimum when $MPIW = 0$, a solution that was found to occasionally appear in practice. This is undesirable as it effectively eliminates the PI altogether. Therefore, Pearce et al. [14] propose that the $MPIW$ be used only in an additive fashion. However, this culminates in an asymmetric magnitude between the two metrics. One may argue, in any case, that the $PICP$ is the more crucial parameter since it establishes the validity of the PIs. Indeed, only once these are found to be valid should an attempt be made to reduce the $MPIW$.
- **Modification to the MPIW:** The $MPIW$ as defined in eq. (5) does not consider if the target values are contained within the PI. Thus, the $MPIW$ in eq. (13) encourages them to shrink even if the target value lies outside the PI bounds to minimize the CWC. This is contrary to the desired behaviour since such PIs should be widened to potentially capture the target value. Instead, the captured MPIW (termed $MPIW_C$), which only includes the MPIW of PIs which contain their target value, is proposed by Pearce et al. [14]. However, this has the adverse effect of requiring more training epochs to converge to a solution since the PIs first need to surround their target values before optimization can truly occur.
- **Scaling of λ with training data size:** Intuitively, a larger sample size instills more confidence in the PICP, and thus λ should be increased to reflect this [13]. This can be problematic for data sets with varying sizes, and indeed, quadrotor applications where data set sizes may vary across flights. Therefore, the sample size may be included directly in the cost function to account for this [14].
- **Non-linearity of the CWC:** The CWC is incompatible with traditional gradient descent training algorithms due to the non-linear and discontinuous properties of the function (in particular, for the PICP) [13]. Consequently, other optimization techniques are typically used, such as simulated annealing [13, 14]. However, this proves to be inconvenient for practical applications since gradient descent methods are the most familiar and popular optimization routine used to train ANNs. However, Pearce et al. [14] point out that commonly used loss functions and activation functions (e.g. ReLU) also exhibit these non-linear and/or discontinuous properties, yet are still compatible with gradient descent methods due to robust software implementations (such as `TensorFlow`) at the cost of convergence guarantees. Moreover, replacing step functions in the CWC (eq. (13)) with their sigmoid counterparts can improve convergence characteristics [14].

These modifications collectively culminate in the 'quality-driven' PI cost function (QD) [14] and is described in eq. (14). The 'soft' PICP, $PICP_s$, is defined in eq. (15) where $\sigma(\cdot)$ denotes the sigmoid function and s is some softening factor. Pearce et al. [14] found that $s = 160$ consistently produces acceptable results across multiple different regression tasks.

$$QD = MPIW_C + N\lambda \max\left(0, (1-\alpha) - PICP_s\right)^2 \tag{14}$$

$$PICP_s = \frac{1}{N}\sum_{i=1}^{N} \sigma(s(y_i - \hat{y}_{L,i})) \cdot \sigma(s(\hat{y}_{i,U} - y_i)) \tag{15}$$

III. Procedures for numerical validation of prediction intervals

To numerically validate the estimated ANN PIs for a quadrotor, a dataset containing multiple, distinct, noisy realizations of a given flight trajectory performed under the same conditions is needed. To this end, the INDI (Incremental Non-linear Dynamic Inversion) controller based quadrotor simulation platform developed by Sun et al. [16] may be employed to generate the mock flight data required under consistent flight conditions. This quadrotor simulation is programmed in MATLAB and Simulink and is intended to simulate the controlled flight of the Parrot Bebop2 quadrotor despite the loss of two opposing propellers wherein control is maintained by virtue of the employed INDI controller. The Parrot Bebop2 quadrotor model employed in the simulation is a rotor-local gray-box polynomial model identified from the previous work of Sun et al. [1, 5, 16, 20, 21]. As such, the simulation platform is capable of simulating a nominal quadrotor whose task is to track a, user-specifiable, position reference signal and/or yaw angle reference. While the simulation itself is programmed in MATLAB, the mock flight data is exported and processed for the numerical validation in Python.

Table 1 Gaussian white noise parameters and statistics used for the two numerical validation schemes. In *Num-Validation-Sim*, the default noise levels (i.e. magnitude of the Power Spectral Density of the white noise) of the quadrotor simulation platform of Sun et al. [16] are used. Likewise, the noise statistics used to generate the white noise vectors for *Num-Validation-Noise* for each sensor are also summarized.

Measurement Variable	**Num-Validation-Sim**	**Num-Validation-Noise**		
	Height of white noise PSD	*Mean*	*Variance*	*Units*
Position	1.00E-08	0	0.1	m
Velocity	1.00E-08	0	0.1	ms^{-1}
Acceleration	1.00E-03	0	0.05	ms^{-2}
Attitude	1.00E-08	0	0.05	rad
Rotational velocity	2.00E-04	0	0.1	$rads^{-1}$
Rotor speeds	0	0	0	RPM

A. Generation of mock-flight data

One way to introduce noise into the simulation measurements is to first obtain a noise-free flight of the quadrotor, after which random white-noise can be added to the relevant inputs and outputs. This approach evaluates the validity of the estimated PIs when subject purely to varying noise. Since the noise statistics of the simulation are controllable, another approach is to instead vary the noise for each flight in the simulation itself. This gives a more realistic representation of the contamination of noise as it propagates through the system. It should be emphasized, however, that the INDI controller used in the simulation also relies on these noisy measurements [16]. Consequently, the control actions produced by the controller will also differ slightly across runs. This contamination will increase the measurement variance across the runs, but is more representative of a true system. Consequently, it is more practical if the estimated PIs are capable of accommodating this as well. Therefore, two different numerical validation experiments are conducted of the quadrotor following the same trajectory.

The first numerical validation experiment - *Num-Validation-Noise* - evaluates the PI validity by making predictions on a number of ($N = 1000$) distinct noisy realizations of the flight for which random white noise is injected to a single, noise-free, flight from the simulation after-the-fact to remove the variance contamination by the controller. Noise is injected into the position, body linear and angular velocity, acceleration, and attitude measurements of stemming from the simulation. The noise statistics used for this are summarized in table 1 and are generated using the `normal` method of `numpy.random.RandomState` in Python. The subsequent power spectral densities (PSDs) of the (virtual) sensor measurements may be found in section VI.A.

A more realistic representation of the variance in inputs and outputs of the system may be obtained by injecting noise directly into the system during the simulation. The simulation platform [16] allows for the specification of the magnitudes of noise for the (virtual) sensor measurements of the quadrotor's position, body linear and angular velocity, acceleration, attitude, and rotor speeds. Therefore, the second numerical validation experiment - *Num-Validation-Sim* - evaluates the PI validity across a number of ($N = 1000$) distinct noisy simulation runs. Note that the noise statistics are not specified directly in the simulation [16], instead the noise is parameterized by the height of its PSD for which Simulink handles the translation of this noise into the system. As such, the default values are used and are summarized in table 1. The corresponding PSDs of these input states are located in the appendix section VI.B.

Therefore, two mock data sets are created for the purposes of PI validation. To be considered valid, the estimated PIs should be wide enough to surround subsequent predictions made by the same models on the noisy data sets up to the confidence criterion (i.e. $PICP \geq 95\%$). Moreover, the estimated PIs should also encapsulate the variation in the target data, again up to this confidence criterion.

B. Simulated input manoeuvres for model identification

Model identification can only occur when adequate excitations are present in the identification data set. Therefore, to excite the forces and moments along all of the axes, input manoeuvres along the specifiable variables (i.e. the x-, y-, and z- positions[‡] and the yaw angle) are conducted. These input manoeuvres occur individually at first, then simultaneously

[‡]Note that, the reference frame used by the simulation is North-East-Down (NED) configuration. Thus, z is negative upwards.

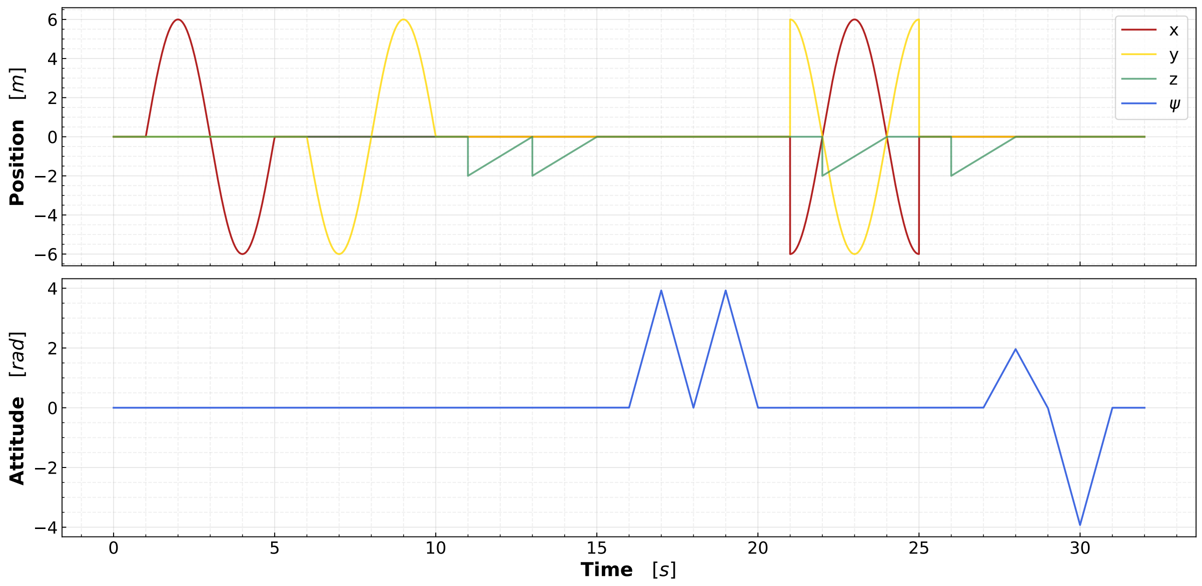


Fig. 1 Manoeuvres conducted in the simulation platform of Sun et al. [16] to generate a flight trajectory that excites all forces and moments. This (mock) data is then used to identify quadrotor models and validate their associated prediction intervals.

to capture some (potential) interaction effects. The magnitudes of these manoeuvres were tuned manually to push the (simulated) quadrotor to its limits while maintaining adequate control since high magnitudes, especially along z, were observed to lead to loss-of-control events. The tuned input manoeuvres are illustrated in fig. 1.

C. Identification of quadrotor models

The procedure for aerodynamic model identification is the same for both *Num-Validation-Noise* and *Num-Validation-Sim*, although separate models are identified for each experiment. An additional flight of the same trajectory (i.e. separate from the $N = 1000$ runs, but with the data set specific noise contamination) is used to identify the ANN models of the aerodynamic forces and moments along with the corresponding PIs using both the bootstrap and quality-driven direct PI estimation methods. This identification flight is further partitioned randomly (once again through `numpy.rand.RandomState`) into training and testing subsets for which 80% of the data is allocated to the training subset. Models are then identified on the training subset through the system identification pipeline developed in **Part II: On outdoor model identification**.

Likewise, the ANN model structures follow exactly from **Part II: On outdoor model identification**. Briefly, the identified ANN quadrotor models are composed of an ensemble of 10 feed-forward ANNs with one hidden layer of 50 neurons. The rectified linear unit (ReLU) is chosen as the hidden layer activation function given its simplicity and success in ANN applications [22]. Furthermore, ADAM is chosen as the optimizer function due to its capabilities. For the bootstrap method, the mean squared error is used as the loss function while eq. (14) is employed for the direct PI method. 150 epochs are used for training with data batch sizes of 400. Moreover, the ANN input vector - and pre-processing of data - is the same as in **Part II: On outdoor model identification**.

As a reference, polynomial models are also identified using the same training data set as for the ANN models, and are subject to the same numerical validation of the PIs. The chosen polynomial model candidates and selection procedure (i.e. step-wise regression) are the same as those defined in **Part II: On outdoor model identification**. Note that these polynomial models target the combined aerodynamic effect of the rotor system, and thus do not model the rotor-local effects as is done in the simulation platform [16]. While the polynomial models primarily serve as a benchmark, the resultant validity of the polynomial PIs also works to help distinguish between invalid ANN-based PIs due to the estimation method and invalid ANN-based PIs due to general identification issues.

IV. Results of numerical validation

Following identification, the ANN (and polynomial) models are then tasked with making predictions on each of the (N = 1000) measurement realizations of the corresponding data sets (i.e. *Num-Validation-Noise* and *Num-Validation-Sim*). The subsequent predictions should lie within the bounds of the originally estimated PIs (i.e. from the identification set) in accordance with the chosen confidence level for such PIs to be considered valid. Here, a confidence level of 95% is taken and, thus, at least 95% of the subsequent predictions should fall within the interval bounds. The $PICP$ directly measures this and is therefore used as an indicator of PI validity. Note, that the $PICP$ with respect to the numerical predictions is different than that with respect to the test data set. The $PICP(Test)$ evaluates the performance of the identified models with respect to a testing subset not seen during training, and therefore relates to the fit of the model and the ability of the PIs to capture variations in the measurement data. In contrast, $PICP(Numerical)$ verifies the ability of the PIs in containing the variance in the models' predictions due to input uncertainty and is therefore related to the consistency of the model predictions. Since the $PICP$ does not consider the PI widths, the $MPIW$ given as a percentage of the range of the target measurement is also computed to verify whether the PIs are practical.

A. Numerical validation through additive noise

The numerical validation results of *Num-Validation-Noise* are summarized in table 2. As an example, fig. 2 illustrates these results for snippet of the identified ANN-bootstrap, ANN-direct PI, and polynomial models of F_z. Shown in grey are the model-specific limits of the N = 1000 noisy predictions (i.e. the bounds of all of the noisy predictions) and are seen to be well-contained within the estimated PIs for all methods. Likewise, the target F_z is also mostly surrounded by the estimated PIs.

Indeed, as is evident from table 2, the polynomial model PIs show valid $PICP(Numerical) \geq 95\%$ for all identified models. Strictly speaking, however, the $PICP(Test) \approx 94\%$ are almost all just below the $PICP \geq 95\%$ threshold and thus do not satisfy the condition. As the $PICP(Test)$ is associated with the containment of test (i.e. unseen) target measurements, this non-fulfilment may be due to over-fitting the training data or potentially unseen inputs in the test subset. Nonetheless, the $PICP(Test)$ for all the polynomial models are close to the $PICP \geq 95\%$ threshold and may be argued to be valid. The associated $MPIW$ for the force models are suitably narrow, but appear to be impractically wide for the moment models. This is likely due to the difficulty faced with the identification of moment models due to the greater prominence of noise and sparsity of excitations. In fact, the power spectral densities of the

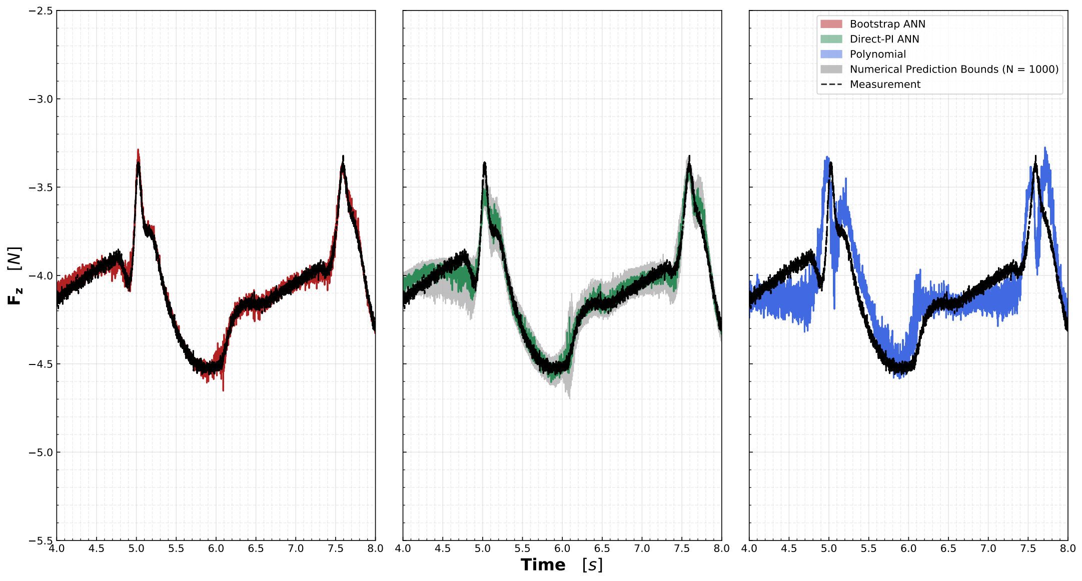


Fig. 2 Comparison of the different ANN PI estimation techniques (i.e. bootstrap and direct PI) as applied to predictions of the F_z model of the simulated quadrotor in [16]. As a reference, the PIs associated with an identified polynomial model of the simulated quadrotor is also shown. In grey are corresponding the limits of the N = 1000 noisy predictions from *Num-Validation-Noise*.

Table 2 Summary of the prediction interval numerical validation results of Num-Validation-Noise. Here, data is derived from a simulated quadrotor flight with noise added after the simulation to the states and outputs to create N=1000 unique noisy realizations. Prediction intervals are estimated on a separate noisy data set using both the Bootstrap and Direct-PI method for the ANNs. As a benchmark, polynomial model prediction intervals are also estimated.

Model	**Bootstrap**			**Direct PI**			**Polynomial**		
	PICP (Test)	*MPIW*	*PICP (Numerical)*	*PICP (Test)*	*MPIW*	*PICP (Numerical)*	*PICP (Test)*	*MPIW*	*PICP (Numerical)*
F_x	97.00	6.00	99.80	98.69	7.63	99.96	94.22	6.48	99.03
F_y	96.52	5.87	99.72	98.90	7.79	99.95	94.34	6.06	98.62
F_z	99.34	3.12	98.87	98.22	5.49	99.98	94.20	5.53	99.02
M_x	94.78	47.80	99.89	87.59	40.63	100.00	94.73	49.33	99.67
M_y	95.22	46.75	99.97	88.14	44.83	100.00	95.03	55.18	99.91
M_z	94.30	51.82	99.99	87.89	40.49	100.00	94.66	49.74	100.00

moment measurements - depicted in fig. 3 - reveals that much of the power in the signal belongs to high-frequency noise. The noise-dominant signal leads to poor moment models and, in practical applications, these should be filtered. However, it is purposefully left unfiltered here in the interest of evaluating PI capabilities of the estimation methods with respect to prominent noise. Indeed, despite this noise, the polynomial models nonetheless boast suitable and valid PIs, demonstrating their robustness. It is desirable that the ANN PI estimation methods also handle such noise-dominant cases well.

Both the bootstrap and directly estimated PIs satisfy the $PICP(Numerical) \geq 95\%$ condition in table 2 for both the force and moment models, and thus successfully capture variations in predictions due to modelling errors. Moreover, both ANN PI estimation methods also satisfy the $PICP(Test) \geq 95\%$ condition for the force models, which supports the validity and reliability of the estimated PIs through these methods for signals with relatively low noise power. However, the direct PI method fails to contain the target data within its PIs for the moment models, given the $PICP(Test) < 95\%$. While this may be an over-fitting issue, it demonstrates the sensitivity of the direct PI method to the noise level in the signal for containing the variations in the targets due to measurement noise. Indeed, the direct PI method holds the narrowest moment model PIs and should widen them to accommodate better $PICP(Test)$ performance. Strictly speaking, the PIs are only valid when both conditions are satisfied. Thus, the direct-PIs can only be considered valid if the noise in the signal does not overshadow the signal itself. Conversely, the bootstrap estimated PIs show a better performance with almost identical $PICP(Test)$ performance as the reference polynomial. As such, the bootstrap estimated PIs may be considered valid and reliable, even with high-noise data. This robustness against noise is in accordance with the conclusions of Khosravi et al. [12] regarding the bootstrap method.

The ANN models mirror the $MPIW$ performances of the polynomial models and accentuate the difficulty in identifying the moment models subject to such high noise contamination. The bootstrap estimated PIs are narrowest for the force models while the directly estimated PIs are the widest. Conversely, the directly estimated PIs are narrowest for the moment models with the polynomial model harboring the widest PIs.

Overall, the results of *Num-Validation-Noise* imply that the ANN PI estimation methods generate PIs which are valid for containing predictions made on noise contaminated input data by satisfying the $PICP(Numerical)$ condition. However, the direct PI method exhibits sensitivity to the power of noise in the data through valid $PICP(Test)$ for the force models (low noise) but not the moment models (high noise). Therefore, it may be argued that the direct PI method estimates valid PIs so long as the dynamics of interest hold greater power than noise in the identification data. In contrast, the bootstrap method produces valid PIs by satisfying both the $PICP(Numerical)$ and $PICP(Test)$ conditions regardless of the power of noise contamination in the identification data. In fact, the bootstrap method almost mirrors the performance of the underlying polynomial, showing great promise for this method.

B. Numerical validation through distinct simulation runs

The results of the *Num-Vaidation-Sim* experiment, where 1000 unique simulation runs are used to numerically validate the estimated PIs, are summarized in table 3. Furthermore, the PSDs of the force and moment measurements

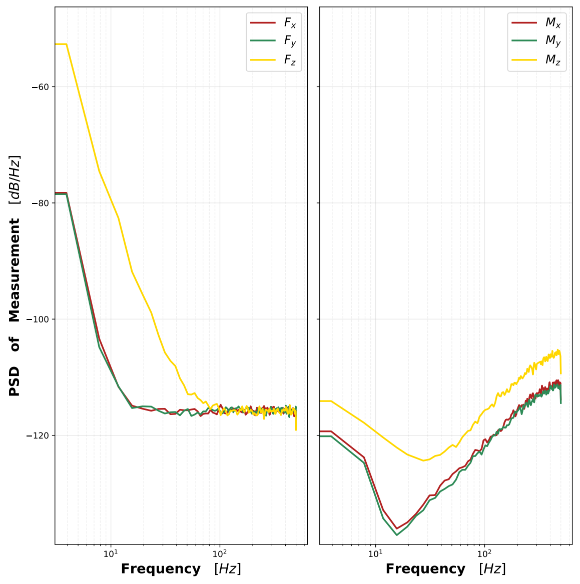


Fig. 3 Power spectral densities (PSD) of the measured force and moments used for model identification in *Num-Validation-Noise*

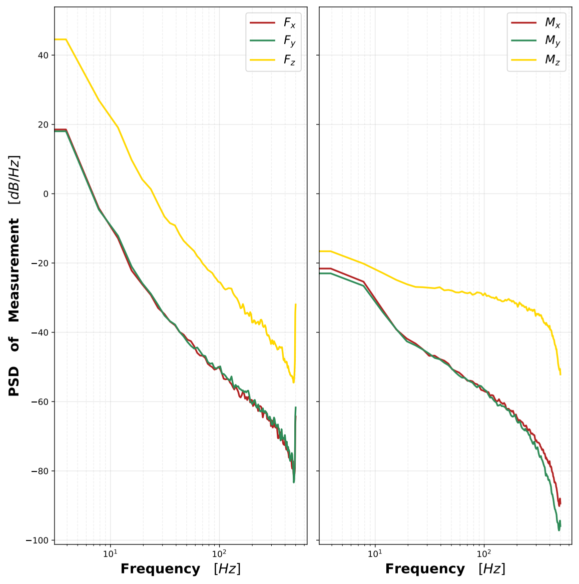


Fig. 4 Power spectral densities (PSD) of the measured force and moments used for model identification in *Num-Validation-Sim*

obtained from the simulation are depicted in fig. 4. From this figure, it is apparent that the noise should not be a significant issue during identification unlike for *Num-Validation-Noise*. Note that a spike at the end of spectra, at 500 Hz, corresponds to the control loop frequency used for the simulation.

The benchmark performance of the polynomial PIs shows valid $PICP(Numerical) \geq 95\%$ for the identified force models with almost valid $PICP(Test) \approx 94\% < 95\%$. This implies that the estimated PIs do well to contain the variation in model predictions due to noisy inputs, but struggle to accommodate variations in the target measurements. Again, this may be a consequence of over-fitting or unseen measurements in the test set. Recall that the prediction interval is only truly descriptive of predictions made on previously seen observations. The associated $MPIW$s of the force models are pragmatically narrow and, consistent with the results of the *Num-Validation-Noise* experiment, the $MPIW$s of the moment models are wider. In contrast, the moment models' $PICP$ performance has deteriorated. Only the M_z model manages valid PIs with $PICP(Test) \geq 95\%$ and $PICP(Numerical) \geq 95\%$. The M_y model exhibits

Table 3 Summary of the prediction interval numerical validation results of Num-Validation-Sim. Here, noisy realizations of the flight are obtained from N=1000 unique runs of the simulation with the same noise statistics. Prediction intervals are estimated on a separate noisy data set using both the Bootstrap and Direct-PI method for the ANNs. As a benchmark, polynomial model prediction intervals are also estimated.

Model	**Bootstrap**			**Direct PI**			**Polynomial**		
	PICP (Test)	*MPIW*	*PICP (Numerical)*	*PICP (Test)*	*MPIW*	*PICP (Numerical)*	*PICP (Test)*	*MPIW*	*PICP (Numerical)*
F_x	99.86	1.58	80.15	98.73	3.99	99.52	94.30	5.66	99.51
F_y	99.13	7.29	79.12	98.55	3.70	97.30	95.92	5.34	99.09
F_z	99.93	4.46	81.46	98.13	6.78	98.96	93.40	6.72	99.44
M_x	99.47	52.63	92.80	99.27	56.92	99.42	70.55	12.20	74.52
M_y	98.07	3.67	32.51	96.66	12.74	91.48	95.13	13.77	89.28
M_z	95.00	17.65	100.00	92.34	18.49	99.79	96.67	19.12	99.87

a valid $PICP(Test) \geq 95\%$ but fails with $PICP(Numerical) < 95\%$. This implies that the variation in measured targets is contained by the PIs, but the variation of predictions is not. This is arguably the worst case scenario since the models' PIs appear to contain the measurement data well but the subsequent predictions are unstable. The extension of the noisy model predictions beyond the PI bounds is evident in fig. 5, which provides a zoomed-in illustrative example of the range of the noisy predictions (in grey) and the estimated PIs of M_y models for each of the estimation techniques. In contrast, the M_x is unable to produce valid PIs with $PICP(Test) < 95\%$ and $PICP(Numerical) < 95\%$ indicating that the associated PIs both fail to capture the target measurements and model predictions on noisy data. However, both the $PICP(Test)$ and $PICP(Numerical)$ are similar (70.55 and 74.52 respectively), perhaps implying a poor model fit.

Similarly, the bootstrap estimated PIs suffer generally poor $PICP(Numerical)$ performance across all models and thus substantially fail to contain the variation in model predictions. Only the M_z model's PIs enjoy a valid $PICP(Numerical) \geq 95\%$. While the PIs of the M_x model are close to the numerical condition, they do not satisfy it ($PICP(Test) = 92.90\% < 95\%$). Despite this, they do outperform the PIs of the corresponding polynomial model, albeit with an increased $MPIW$ (Bootstrap: $MPIW = 52.63$, Polynomial: $MPIW = 12.20$). However, for other models $PICP(Numerical) \approx 80\%$ with a particularly poor performer in the M_y ANN bootstrap model ($PICP(Numerical) = 32.51\% < 95\%$). This is unacceptable as accounting for model prediction variances is one of the fundamental purposes of using the PIs, and indicates that model predictions are unstable and routinely extend beyond the estimated PI bounds. This is clearly observable in fig. 5 where the noisy prediction bounds (in grey) appear to surround the estimated PIs, which is exactly contrary to the desired behavior. This suggests that the bootstrap PIs are invalid. Peculiarly, the bootstrap PIs appear to surround the measurement data well $PICP(Test) \geq 95\%$ for all identified models, implying a good model fit. Such a result accentuates concerns pertaining to over-fitting and the reliability of ANN model predictions despite acceptable model accuracy. The associated $MPIW$s are comparable to, if not narrower than, the polynomial models with the exception of the aforementioned wide ANN bootstrap M_x PIs.

Likewise, the direct PI method manifests similar $MPIW$ as the polynomial model, again with the exception of the M_x. However, in contrast to the bootstrap and polynomial PIs, the directly estimated PIs are found to be mostly valid. Only the ANN direct PI M_z and M_y do not meet the conditions of $PICP(Test) = 92.34\% < 95\%$ and $PICP(Numerical) = 91.48\% < 95\%$ respectively. Indeed, from fig. 5, the direct PI ANN visually exhibits the most successful PIs in containing both the target measurement and the variation in numerical predictions. Aside from the direct

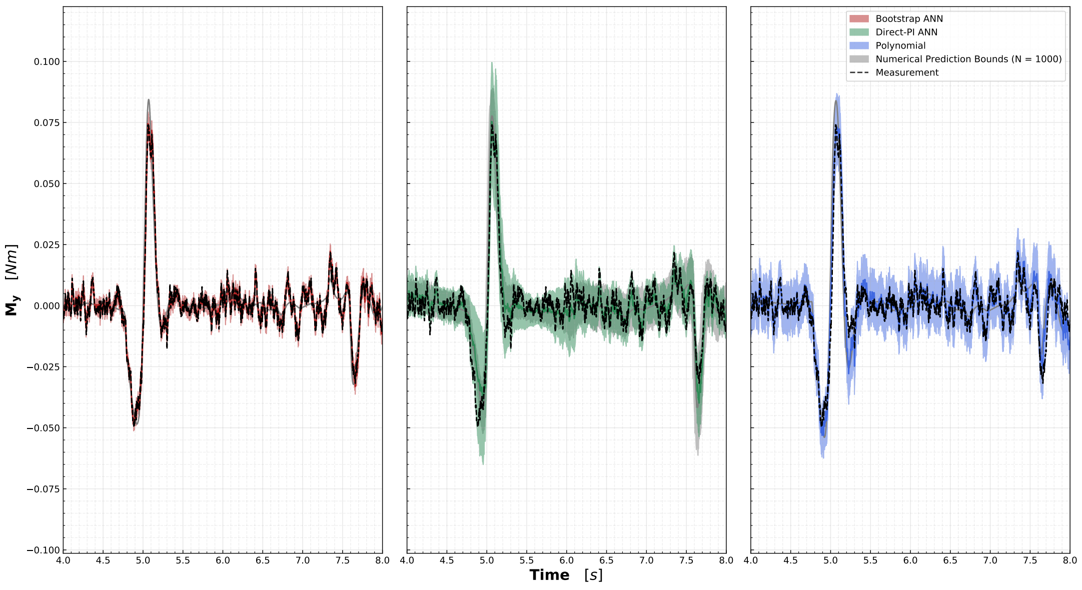


Fig. 5 Comparison of the different ANN PI estimation techniques (i.e. bootstrap and direct PI) as applied to predictions of the M_y model of the simulated quadrotor in [16]. As a reference, the PIs associated with a polynomial model of the simulated quadrotor is also shown. In grey are corresponding the limits of the $N = 1000$ noisy predictions from *Num-Validation-Sim*. These should lie mostly within the model prediction intervals for valid PIs.

PI M_z and M_y models, all other PICP conditions are met with $PICP(Test) \geq 95\%$ and $PICP(Numerical) \geq 95\%$. The valid PIs found for the force models of direct PI method are similar to that of the underlying polynomial with comparable $MPIWs$. All models clearly struggle with the identification and estimation of the moment model PIs. However, the direct PI method sports arguably the most reliable PIs and outperforms the bootstrap method in this regard.

In general, the results of *Num-Validation-Sim* show that PI performance has deteriorated for the polynomial and bootstrap estimated PIs in comparison to *Num-Validation-Noise*. Surprisingly, the direct PI estimation method manages to improve performance over the previous experiment. A potential explanation for this is that, as aforementioned, the controller of the simulation utilizes the noisy measurements and therefore outputs different control actions across the simulation runs. Both the bootstrap ANN and polynomial PI estimation methods assume a normal distribution of prediction errors. With the additional variations induced by the controller, this assumption no longer holds. Hence, the decay in performance of these methods is likely a consequence of this. Moreover, for the bootstrap method specifically, the poor $PICP(Numerical)$ performance may be a byproduct of too few ANNs in the ensemble since this is directly related to estimates of model uncertainty and variation. However, note that the direct PI method also harbors the same amount of ANN ensembles, but does not suffer poor $PICP(Numerical)$ performance. Moreover, the direct PI method is more robust to the additional variation induced by the controller since it does not rely heavily on the assumption that the prediction errors are normally distributed. Instead, the PIs are directly obtained through the optimization of the PI metrics (i.e. PICP and MPIW) in the ANN training cost function (refer to eq. (14)). This characteristic is more desirable for practical applications where errors may not only arise from white noise.

For both numerical validation experiments, all estimated PIs (i.e. polynomial, bootstrap, and direct PI) appear to struggle with containing either the measured targets or the variation in predictions when they struggle to fit the target data, as seen for the moment models. This may be a consequence of the measurement data itself, which for the moment models sees sparse excitations and a relatively high noise power. In contrast, when the underlying models are capable of describing the data well, such as for the force models, the PIs estimated from the bootstrap and direct PI methods are found to be numerically valid and reliable.

V. Conclusion

Through mock data obtained from a high-fidelity quadrotor simulation, both the bootstrap and quality-driven direct PI estimation techniques for ANNs are shown to be numerically valid for identified quadrotor models, assuming that a few constraints are satisfied.

The bootstrap method is found to be robust against noise and closely mirrors the performance of equivalent polynomial PIs. As with the polynomial PIs, the bootstrap method relies heavily on the assumption that the prediction errors are normally distributed. When this is not the case, the bootstrap estimated PIs fail to enclose the variation in model predictions due to uncertainty in the inputs. Indeed, only when the prediction errors are normally distributed are the bootstrap PIs found to be valid. Future research should therefore investigate the extent of this dependence by varying the number of ensembles used for identifying the quadrotor model as this is directly related to the approximation of the modelling uncertainty.

Contrarily, the quality-driven direct PI estimation method is found to be less sensitive to the assumption of normally distributed errors and is capable of producing numerically valid PIs even when this assumption is not satisfied. However, the quality-driven direct PI method is shown to be sensitive to the level of noise in the inputs. When the power of noise is low relative to the interested dynamics, then the quality-driven direct PI estimation method successfully estimates valid and reliable PIs. However, if the level noise is high, then the PIs fail to encompass the variation in the target data but manage to encompass the variation in model predictions. Subsequently, the estimated PIs are invalid for high-noise data. It is recommended that future work investigates the noise sensitivity of the direct PI method to determine clear requirements for the identification data in order to obtain valid PIs.

Further recommendations include validating other ANN PI estimation techniques, such as the LUBE method, and running more epochs for numerical validation to improve confidence in the results. Although difficult to do practically, it is desirable to also validate these PI estimation techniques using real flight data in the future through repeated flights of a quadrotor in a controlled environment. Nevertheless, the results of this paper show promise for the numerically validated PI estimation techniques, and defines some rudimentary guidelines for ensuring their reliability.

References

[1] Sun, S., de Visser, C., and Chu, Q., "Quadrotor Gray-Box Model Identification from High-Speed Flight Data," *Journal of Aircraft*, Vol. 56, No. 2, 2019, pp. 645–661. https://doi.org/10.2514/1.C035135, URL https://doi.org/10.2514/1.C035135.

[2] Hoffmann, G., Huang, H., Waslander, S., and Tomlin, C., "Quadrotor helicopter flight dynamics and control: Theory and experiment," *AIAA guidance, navigation and control conference and exhibit*, 2007, p. 6461. https://doi.org/https://doi.org/10.2514/6.2007-6461.

[3] Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., and Kumar, V., "Influence of aerodynamics and proximity effects in quadrotor flight," *Experimental robotics*, Springer, 2013, pp. 289–302. https://doi.org/https://doi.org/10.1007/978-3-319-00065-7_21.

[4] Mahony, R., Kumar, V., and Corke, P., "Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor," *IEEE Robotics Automation Magazine*, Vol. 19, No. 3, 2012, pp. 20–32. https://doi.org/10.1109/MRA.2012.2206474.

[5] Sun, S., Schilder, R. J., and de Visser, C. C., *Identification of Quadrotor Aerodynamic Model from High Speed Flight Data*, 2018. https://doi.org/10.2514/6.2018-0523, URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0523.

[6] Bansal, S., Akametalu, A. K., Jiang, F. J., Laine, F., and Tomlin, C. J., "Learning quadrotor dynamics using neural network for flight control," Institute of Electrical and Electronics Engineers Inc., 2016, pp. 4653–4660. https://doi.org/10.1109/CDC.2016.7798978.

[7] Mohajerin, N., Mozifian, M., and Waslander, S., "Deep Learning a Quadrotor Dynamic Model for Multi-Step Prediction," Institute of Electrical and Electronics Engineers Inc., 2018, pp. 2454–2459. https://doi.org/10.1109/ICRA.2018.8460840.

[8] Bauersfeld, L., Kaufmann, E., Foehn, P., Sun, S., and Scaramuzza, D., "NeuroBEM: Hybrid Aerodynamic Quadrotor Model," *CoRR*, Vol. abs/2106.08015, 2021. URL https://arxiv.org/abs/2106.08015.

[9] Lu, S., and Başar, T., "Robust nonlinear system identification using neural-network models," *IEEE Transactions on Neural Networks*, Vol. 9, 1998, pp. 407–429. https://doi.org/10.1109/72.668883.

[10] Chen, S., Billings, S. A., and Grant, P. M., "Non-linear system identification using neural networks," *International Journal of Control*, Vol. 51, 1990, pp. 1191–1214. https://doi.org/10.1080/00207179008934126, URL https://www.tandfonline.com/doi/abs/10.1080/00207179008934126.

[11] Schumann, J., Gupta, P., and Nelson, S., "On verification and validation of neural network based controllers," *In Proceedings, International Conference on Engineering Applications of Neural Networks*, 2003, p. 47.

[12] Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F., "Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances," *IEEE Transactions on Neural Networks*, Vol. 22, No. 9, 2011, pp. 1341–1356. https://doi.org/10.1109/TNN.2011.2162110.

[13] Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F., "Lower Upper Bound Estimation Method for Construction of Neural Network-Based Prediction Intervals," *IEEE Transactions on Neural Networks*, Vol. 22, No. 3, 2011, pp. 337–346. https://doi.org/10.1109/TNN.2010.2096824.

[14] Pearce, T., Zaki, M., Brintrup, A., and Neely, A., "High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach," , 2018. URL https://arxiv.org/abs/1802.07167.

[15] Pearce, T., Leibfried, F., Brintrup, A., Zaki, M., and Neely, A., "Uncertainty in Neural Networks: Approximately Bayesian Ensembling," , 2020. URL https://arxiv.org/abs/1810.05546.

[16] Sun, S., Wang, X., Chu, Q., and d. Visser, C., "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors," *IEEE Transactions on Robotics*, 2020, pp. 1–15. https://doi.org/10.1109/TRO.2020.3010626.

[17] Heskes, T., "Practical Confidence and Prediction Intervals," *Advances in Neural Information Processing Systems 9*, MIT press, 1997, pp. 176–182.

[18] Davison, A. C., and Hinkley, D. V., *Bootstrap Methods and their Application*, Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 1997. https://doi.org/10.1017/CBO9780511802843.

[19] Efron, B., and Tibshirani, R., *An Introduction to the Bootstrap*, Chapman & Hall/CRC Monographs on Statistics & Applied Probability, Taylor & Francis, 1994. URL https://books.google.nl/books?id=gLlpIUxRntoC.

[20] Sun, S., Sijbers, L., Wang, X., and de Visser, C., "High-Speed Flight of Quadrotor Despite Loss of Single Rotor," *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, 2018, pp. 3201–3207. https://doi.org/10.1109/LRA.2018.2851028.

[21] Sun, S., and de Visser, C., "Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime," *IEEE Robotics and Automation Letters*, Vol. 4, No. 4, 2019, pp. 3868–3875. https://doi.org/10.1109/LRA.2019.2928758.

[22] Lecun, Y., Bengio, Y., and Hinton, G., "Deep learning," *Nature*, Vol. 521, 2015, pp. 436–444. https://doi.org/10.1038/nature14539.

VI. Appendix

In this appendix, supplementary plots associated with the numerical validation of the estimated prediction intervals are shown.

A. Input noise spectra for Num-Validation-Noise

In this subsection, supplementary plots regarding the power spectral densities of the noise contaminated input states are shown to contextualize the noise level for *Num-Validation-Noise*. Here, random white noise is added $N = 1000$ times to a clean data set from the quadrotor simulation of Sun et al. [16].

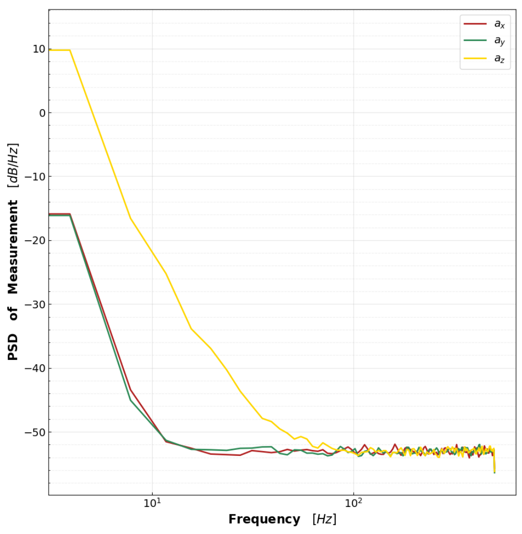


Fig. 6 Power spectral density of the simulated noisy acceleration measurements for *Num-Validation-Noise*

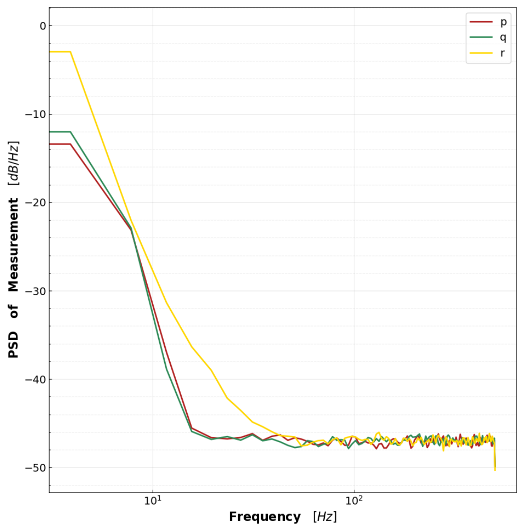


Fig. 7 Power spectral density of the simulated noisy rotational rates measurements for *Num-Validation-Noise*

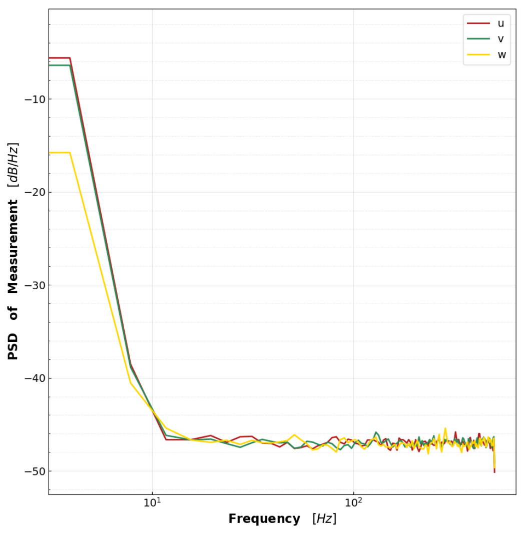


Fig. 8 Power spectral density of the simulated noisy velocity measurements for *Num-Validation-Noise*

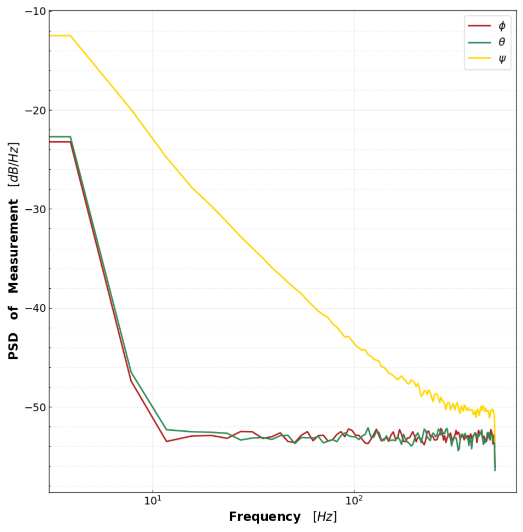


Fig. 9 Power spectral density of the simulated noisy attitude measurements for *Num-Validation-Noise*

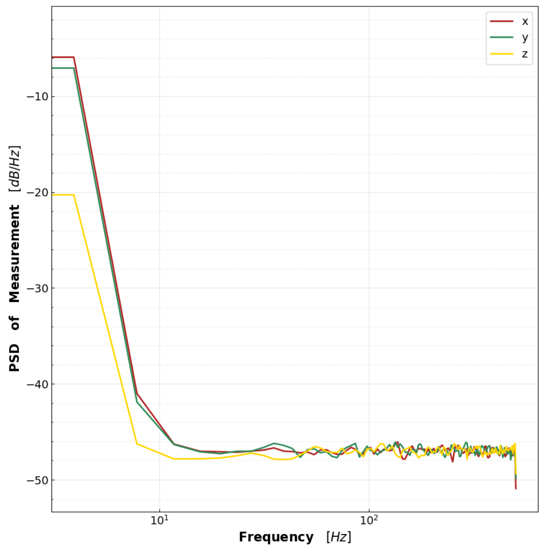


Fig. 10 Power spectral density of the simulated noisy position measurements for *Num-Validation-Noise*

B. Input noise spectra for Num-Validation-Sim

In this subsection, supplementary plots regarding the power spectral densities of the noise contaminated input states are shown to contextualize the noise level for *Num-Validation-Sim*. Here, $N = 1000$ noisy simulations of the same flight trajectory are flown by a quadrotor in simulation of Sun et al. [16].

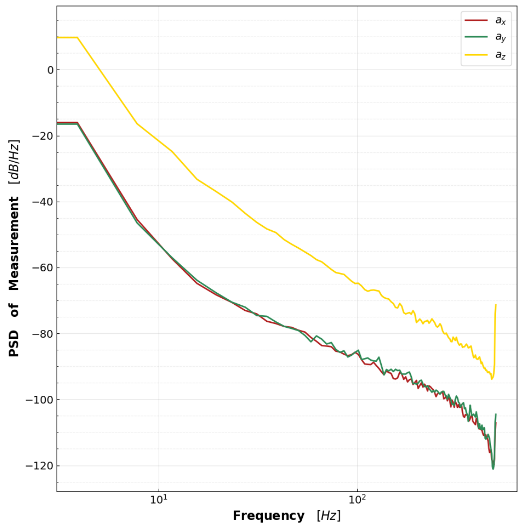


Fig. 11 Power spectral density of the simulated noisy acceleration measurements for *Num-Validation-Sim*

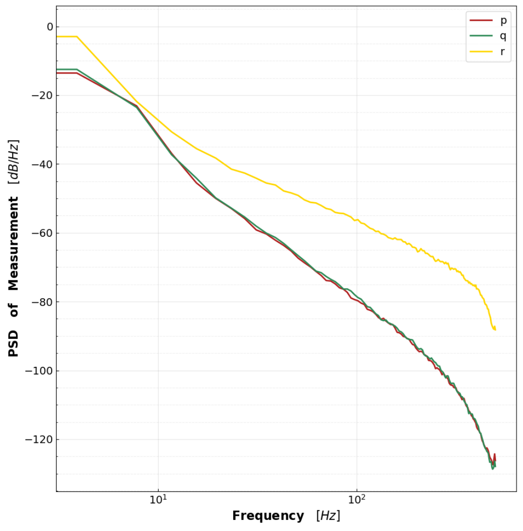


Fig. 12 Power spectral density of the simulated noisy rotational rates measurements for *Num-Validation-Sim*

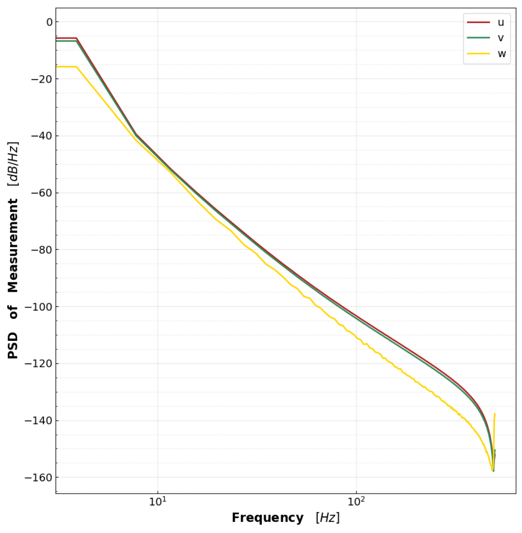


Fig. 13 Power spectral density of the simulated noisy velocity measurements for *Num-Validation-Sim*

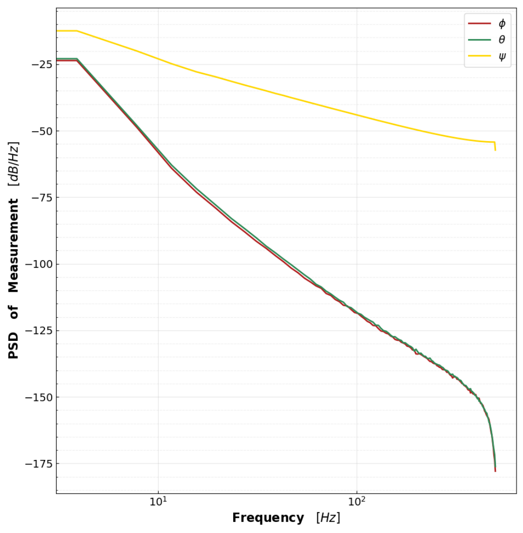


Fig. 14 Power spectral density of the simulated noisy attitude measurements for *Num-Validation-Sim*

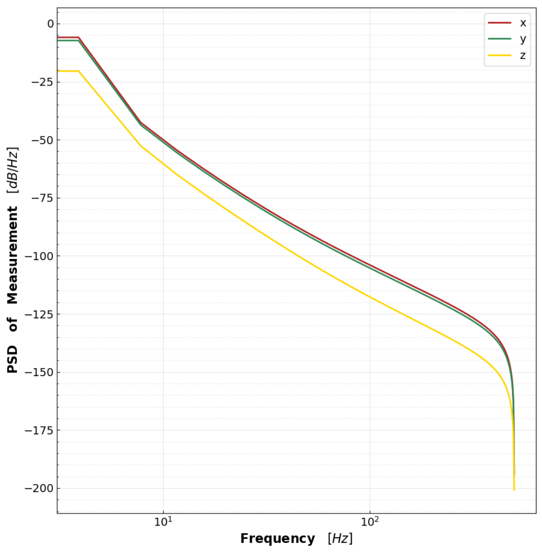


Fig. 15 Power spectral density of the simulated noisy position measurements for *Num-Validation-Sim*

Part II

Scientific Paper: On outdoor model identification

Development of a system identification routine suitable for modelling high-speed and aggressive quadrotor flight outdoors

J.J. van Beers [*] and Dr. ir. C.C. de Visser [†]
Delft University of Technology, 2629 HS Delft, The Netherlands

As research endeavours and commercial applications demand more of the quadrotor, it is only natural to develop models which can facilitate this. Currently, analytical descriptions of the quadrotor are rudimentary and most data-driven quadrotor models are identified from flight data collected indoors. Therefore, existing quadrotor models are constrained to a narrow region of the flight envelope due to the inherent restrictions imposed by these indoor spaces. In contrast, the flexibility afforded by outdoor spaces facilitates the exploration of the majority of the quadrotor's flight envelope. To enable the development of high fidelity quadrotor models, a modular system identification pipeline that is compatible with outdoor high-speed flight and aggressive manoeuvring is created in the present work. Drawing inspiration from current state-of-the-art quadrotor models, polynomial step-wise regression, artificial neural networks (ANNs), and a hybrid approach fusing the merits of the two techniques are implemented in the pipeline. Through this pipeline, for the first time, high fidelity quadrotor models which accurately capture high-speed flight of up to $19\ ms^{-1}$ and aggressive manoeuvres such as punch-outs, flips, and barrel rolls, are identified from real outdoor flight data despite the contamination of unknown wind. As a proxy for reliability, the confidence of the identified models' predictions are encapsulated through accompanying prediction intervals (PIs). The subsequent validity of the identified models is evaluated by injecting these models into a quadrotor simulation to examine their responses to simple attitude step inputs. Indeed, the contamination of wind is evident in the models' simulated responses through non-zero constant force biases parallel to the incident wind. Nonetheless, these simulations show promise for the developed system identification pipeline given that the simulated models faithfully reproduce the forces and moments observed in the measurement data when subject to similar attitude inputs. Contrary to expectations, the simulations demonstrate that the polynomial model consistently produces the most feasible and useful models. The dense architecture of the employed ANNs, including those for the hybrid approach, appear to promote and propagate instabilities arising from the wind contamination. The associated PIs are found to grow in tandem with these instabilities and increasing uncertainty in the system, accentuating the utility of such PIs.

Contents

[*]Student, Faculty of Aerospace Engineering: Control and Simulation
[†]Assistant professor, Faculty of Aerospace Engineering: Control and Simulation

I. Introduction

QUADROTORS are highly autonomous and versatile yet simple, cheap, and easy to maintain [1, 2]. Accordingly, the quadrotor platform is popular in both the commercial sector and as an outlet for research in the fields of robotics and aerospace. Such extensive research into this platform has culminated in numerous advancements from demonstrations of autonomous flight with aggressive manoeuvring [3–6] to the recent extension of quadrotor models to the high speed regime [7–11].

Peculiarly, the first principle model of the quadrotor, which sees frequent use in contemporary research, is invalid for the majority of its operating flight envelope including high-speed flight and aggressive manoeuvring [12]. Research conducted in these regions of the flight regime are often facilitated by high sensor update rates and the capabilities of the employed controllers. It comes as no surprise, then, that a considerable body of literature is invested in developing more compelling controllers and, conversely, research into quadrotor models sees scarce attention. For instance, quadrotors which autonomously complete aggressive manoeuvres both indoors [3, 5] and outdoors [6] are often explicitly trained on a specific set of manoeuvres and thus subsequent (learned) policies or controllers are incompatible with different manoeuvres, even though the underlying dynamics may be the same. To this end, current autonomous aggressive flight is rather inflexible and restrictive.

Instead, through high fidelity quadrotor aerodynamic models, controllers may exploit knowledge of the dynamics during aggressive manoeuvring to maintain and even enhance performance across a larger domain of the flight envelope [11]. In fact, many controllers - including sensor-based controllers such as INDI [13] - depend in some way on accurate quadrotor models. As observed by Molenkamp et al. [4], significant discrepancies between the quadrotor internal model and true system often leads to crashes, especially during aggressive manoeuvring. Furthermore, accurate models of the quadrotor may be deployed in simulation not only to safely prototype new controller schemes, but also as a training environment for hobbyist or professional pilots. Therefore, high fidelity models of the quadrotor only seek to extend the capabilities and safety of the quadrotor.

Since existing analytical models are limited in their capacity to describe the complex aerodynamic phenomena prevalent in much of the quadrotor's flight envelope, many turn to data-driven system identification techniques to obtain high fidelity quadrotor models. For instance, Sun et al. [7, 8, 10] have repeatedly demonstrated the success of using stepwise polynomial regression to identify high-speed 'gray-box' quadrotor models from a set of candidate regressors. The term 'gray-box' stems from the fact that the resultant polynomial structures, and the order in which regressors are selected, provide insight into what (combinations of) states are significant for the underlying dynamics. However, the candidate regressors themselves may not all necessarily represent anything physically which obscures such physical interpretations. Moreover, artificial neural networks (ANNs) have also been shown to improve upon quadrotor model accuracy [11, 14–16] by virtue of their generalizability [14] and aptitude for capturing unknown non-linearities [17, 18]. However, as a black-box modelling technique, it is ambiguous as to how exactly the inputs interact to produce the model outputs. Indeed, some literature (e.g. [19, 20]) discusses methods for assessing input importance for ANNs in which parallels to input sensitivity analyses may be drawn. For example, the 'input weight' method which associates the relative importance of a given (normalized) input with the corresponding connection weight [19].

Currently, a substantial amount of quadrotor aerodynamic models are developed using data collected indoors (e.g. [8, 10, 11, 21]), in conditions devoid of external (potentially non-stationary) disturbances, and where accurate external motion capturing systems are available to aid state estimation. Such luxuries are absent in outdoor applications, where the majority of commercial quadrotors operate. Even the high-speed models of Sun et al. [8] were identified from flights conducted in a wind-tunnel, also equipped with an external motion capturing system, and therefore do not consider the effects of out-of-plane disturbances (e.g. gusts) during high-speed flight. Moreover, indoor spaces impose restrictions on the achievable speeds and manoeuvres of the quadrotor. For instance, sudden bursts of thrust (known as 'punches'[*] or 'punch out' for purely vertical motion) are difficult - if not impossible - to conduct indoors. Consequently, much of the quadrotor's flight envelope remains unmodelled.

It is clear that the development of high fidelity quadrotor aerodynamic models may be expediated through the transition to outdoor flight. However, the challenge with identifying such models in outdoor applications lies in the determination of the velocity of the quadrotor solely from on-board sensors compounded with modelling non-stationary effects while rejecting disturbances introduced into the system by the environment. Though airspeed sensors, such as the Mateksys ASPD-4525[†], do exist they have yet to be validated on a quadrotor. Currently, velocity information is most readily accessible from a GPS-module. However, such velocity estimates are infrequent[‡] and only measure the

[*]For an example, see manoeuvres starting at 15:50 here: `https://www.youtube.com/watch?v=Ia2N9fep84w`

[†]See: `http://www.mateksys.com/?portfolio=aspd-4525`

[‡]Exact update rates vary depending on the module used, but for common sensors such as the TBS M8.2, this is around $1 - 10$ Hz.

ground speed of the quadrotor whereas the dynamics which influence the quadrotor depend on the air speed. Moreover, non-stationary effects are ideally accounted for by an online, or otherwise adaptive, system identification routine.

However, before a functional online system identification routine may be realized many conceptual and practical issues first need to be ironed out, including the successful (offline) identification of outdoor quadrotor models which are suitable for high-speed and aggressive flight. Accordingly, a system identification pipeline is developed in this paper to facilitate such outdoor model identification. Indeed, it is unknown what system identification techniques are even suitable for outdoor modelling. Therefore, the computational complexity of the algorithms is of minor importance at this stage as the main focus is on determining suitable techniques for outdoor model identification. Hence, given the abundance of potential system identification methods, the developed system identification pipeline is designed to be modular. To illustrate this functionality, three different system identification methods are deployed for model identification. Given the modelling successes of both the step-wise regression algorithm [8] and ANNs [11, 16], both of these techniques are employed here for comparison. The final approach implemented in the pipeline fuses the two techniques in the so-called hybrid approach, which is novel to this paper.

However, these techniques alone do not provide any insight on the reliability of their subsequent predictions. Ensuring safety and reliability are paramount to the accepted use of autonomous quadrotors, especially for outdoor operations. Therefore, the models identified in this paper are accompanied by prediction intervals (PIs) which provide insight into the associated reliability of a models' prediction. For the polynomial models, there exist analytical formulations for obtaining these PIs. While this is not the case for ANN models, recent work has numerically validated a few ANN PI estimation techniques for ANN quadrotor models (see **Part I: On prediction intervals**). Among these, the quality-driven direct PI method is chosen [22] as it does not rely on the assumption that the prediction errors are normally distributed, which may be the case for outdoor flight (e.g. due to the presence of wind). While this method is sensitive to noise, the identification data used here is not heavily contaminated with noise.

The models themselves are identified on real flight data gathered both indoors and outdoors, using various quadrotor platforms. To limit the scope of this paper, only one quadrotor platform - the MetalBeetle - is discussed in depth here since it is suitable for outdoor flight. In these outdoor flights, the MetalBeetle achieves velocities of up to 19 ms^{-1}. Indeed, while the models' performance may be evaluated with respect to their ability to reproduce the aerodynamic forces and moments, it does not alone illustrate how useful the identified models actually are. Therefore, the identified models are injected into a quadrotor simulation to evaluate their utility and the feasibility of their responses to common inputs, such as a step attitude command. Literature on quadrotor model identification often only assess the model accuracy and neglect this additional step. As will be shown here, the simulation results provide additional insight into the utility of the identified models.

Therefore, the contributions of this research include (i) the development of a modular system identification pipeline suitable for the identification of outdoor quadrotor models including high-speed and aggressive flight (ii) a novel hybrid approach fusing the merits of the polynomial and ANN system identification techniques (iii) a simulation environment in which the identified models' utility may be evaluated.

Before any of these contributions are presented, a first principle model of the quadrotor is introduced with subsequent analytical extensions from literature in section II. These analytical models serve as inspiration for the construction of model input variables and regressors. A brief overview of the MetalBeetle is also given in this section. Subsequently, the implemented system identification methods (i.e. polynomial step-wise regression, ANN, and hybrid) and pipeline itself are summarized in section III. The approaches to data collection and processing are explained in section IV. Subsequently, section V presents the results of the model identification of the MetalBeetle while section VI describes the model structures, potential sensitivities, and presents the simulation results.

II. Fundamentals of the quadrotor platform

Preliminary first principle and analytical models of the quadrotor, whose structures motivate the subsequent selection of model input variables, are presented in this section. To facilitate this description, the employed reference frames are defined first. This section concludes wit the quadrotor platforms used for both indoor and outdoor flight data collection are also summarized here.

A. Quadrotor reference frames

Before discussion models of the quadrotor itself, two reference frames are first defined in order to describe the motion of the quadrotor in space. Subsequent models are then derived in the context of these reference frames. The inertial reference frame, denoted by $\{E\} = \{O_E, x_E, y_E, z_E\}$, is fixed to a point on the ground and is oriented using the north-east-down (NED) configuration. To describe the orientation of the drone with respect to the inertial frame, the body reference frame, $\{B\} = \{O_B, x_B, y_B, z_B\}$, is defined and is fixed to the quadrotor body with origin, O_B, at its center of gravity, x_B pointing forwards, y_B to the right, and z_B aligned with gravity when the drone is hovering as illustrated in fig. 1. The superscript B in A^B indicates that the entity A is expressed in the body frame, $\{B\}$ and a lack of superscript specifies that A is expressed in the inertial frame, $\{E\}$, unless stated otherwise. The body frame can be related to the inertial frame through the euler angles: ϕ (roll), θ (pitch), and ψ (yaw).

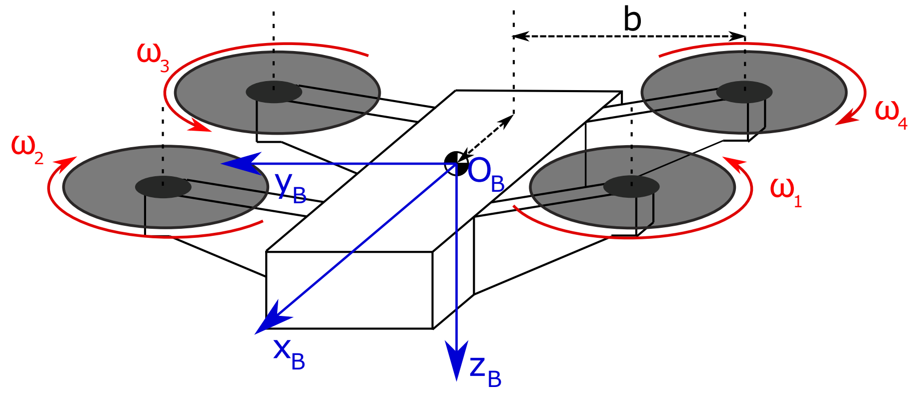


Fig. 1 Illustration of the generic quadrotor in the breaststroke configuration, whereby rotor one is rotating counterclockwise, denoted by ω_1. Depicted is the body reference frame, B, with origin, O_B, at the quadrotor's center of gravity (c.g.). Also shown are geometric parameters b and ℓ, which respectively represent the y_B and x_B distances from the rotor centers of rotation to the c.g. of the quadrotor.

B. Simple model of the quadrotor

Let the position of the quadrotor in the inertial frame be given by $\boldsymbol{\xi} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ and velocity by $\mathbf{V} = \begin{bmatrix} u & v & w \end{bmatrix}^T$. Assuming that the quadrotor is a rigid body, a simple model of the quadrotor can be derived by following Newton's laws of motion. The resultant force and moment equations of motion are given by eq. (1) and eq. (2) respectively.

$$m\dot{\mathbf{V}} = m\mathbf{g} + R_{EB}\mathbf{F}^B \tag{1}$$

$$\mathbf{I_v}\dot{\mathbf{\Omega}}^B + \mathbf{\Omega}^B \times \mathbf{I_v}\mathbf{\Omega}^B = \mathbf{M}^B \tag{2}$$

In these equations, m denotes the mass of the quadrotor and $\mathbf{I_v}$ gives its moment of inertia. The gravity vector along z_E is given by $\mathbf{g}$. The resultant force acting on the quadrotor, $\mathbf{F}^B$, is a sum of the control forces exerted by the rotors and the aerodynamic forces acting on the quadrotor. Since these forces act on the quadrotor itself, they need to be transformed into the inertial reference frame for which R_{EB} represents the corresponding rotation matrix. Consequently, eq. (1) is defined in the inertial frame. Equivalently, the force equation may be expressed in $\{B\}$ through eq. (3) where R_{BE} denotes the rotational matrix from $\{E\}$ to $\{B\}$. The rotational rates of the quadrotor are given by $\mathbf{\Omega}^B = \begin{bmatrix} p & q & r \end{bmatrix}^T$ and are expressed in the body frame. As such, the total moment acting on the quadrotor, $\mathbf{M}^B$, is also defined in the body frame and is comprised of the control moments, aerodynamic moments, and gyroscopic moments (due to the rotation of the rotors).

$$m\left(\dot{\mathbf{V}}^B + \mathbf{\Omega}^B \times \mathbf{V}^B\right) = R_{BE}m\mathbf{g} + \mathbf{F}^B \tag{3}$$

Note that $\mathbf{V}^B$ (and equivalently $\mathbf{V}$) represents the relative airspeed of the quadrotor, and therefore depends on the wind speed, $\mathbf{V_W}$. Much of the research into quadrotor dynamics and control is conducted under windless conditions (see, for instance, [3, 9, 23–25]). While this is perhaps suitable for indoor applications (such as drone racing), ignoring wind is problematic for outdoor applications where such effects are typically non-negligible (such as off-shore wind turbine inspection). Assuming that the instantaneous wind vector is known in $\{E\}$, the airspeed of the quadrotor can be expressed through eq. (4) [8]. Here, $\mathbf{V_G}$ denotes the ground speed of the quadrotor.

$$\mathbf{V} = \mathbf{V_G} - \mathbf{V_W} \tag{4}$$

In a conventional quadrotor, the control forces and moments can be manipulated by controlling the thrust of individual rotors. A simple model of the total thrust generated by the rotors is given by eq. (5) where ω_i denotes the angular velocity of the i^{th} rotor[†] and κ_0 is a constant for the quadrotor which represents properties of the rotor and air density [7, 26, 27]. κ_0 is typically identified through measurements of the hovering quadrotor in windless conditions where aerodynamic effects are considered negligible [27]. Note that, per the definition of the body reference frame, the thrust, T, acts along the negative z_B direction.

$$T = \kappa_0 \sum \omega_i^2 \tag{5}$$

The rolling, pitching, and yawing control moments are given by eqs. (6) to (8) respectively where $S_R = -1, 1$ denotes the rotation direction of rotor 1. A counterclockwise rotation is denoted by -1 and clockwise rotation is represented by 1. Thus, following the diagram of fig. 1, $S_R = -1$. Note that the definitions of the control moments vary between different quadrotor configurations, depending on their rotor numbering and rotation configurations.

$$U_p = (\omega_1 + \omega_4) - (\omega_2 + \omega_3) \tag{6}$$

$$U_q = (\omega_1 + \omega_2) - (\omega_3 + \omega_4) \tag{7}$$

$$U_r = S_R \left[(\omega_1 + \omega_3) - (\omega_2 + \omega_4)\right] \tag{8}$$

The magnitudes of the control moments exerted by the quadrotor depends on its geometry (i.e. arm length to the quadrotor's center of gravity), the thrust coefficient, κ_0, (for U_p and U_q) and the rotor torque coefficient, τ_0 (for U_r). Following fig. 1, the arm lengths are b (along the y_B axis) and ℓ (along the x_B axis). As with κ_0, τ_0 may be identified during hover under windless conditions. Thus, a simple moment model may be obtained through

$$\mathbf{M}_c^B = \begin{bmatrix} b\kappa_0 U_p \\ \ell\kappa_0 U_q \\ \tau_0 U_r \end{bmatrix}$$

C. Analytical extensions to the quadrotor model

The simple quadrotor model does not desrcibe how the different quadrotor states, and the interactions therein, influence the aerodynamic forces experienced during flight. As such, analytical extensions to this simple model have been developed in literature to detail some of the components which constitute the resultant aerodynamic force.

1. Thrust variation

One of the prominent aerodynamic effects ubiquitous at higher velocities is the increased dependence of thrust - and therefore altitude control - on the relative velocity between the rotor and surrounding air, V (see eq. (4)), and the angle of attack of the rotor disks, α_r [2, 28]. For simplicity, it is often assumed that the rotor disk plane is in line with the $x_B - y_B$ plane.

Air gains additional energy, through an increased velocity, as it passes through the rotor. Let this additional velocity be the induced velocity, v_{in}. The thrust of the quadrotor, at arbitrary α_r, can be found using momentum theory and is summarized by eq. (9) [27].

$$T = 2\rho A v_{in} \sqrt{V^2 + 2V\sin(\alpha_r) v_{in} + v_{in}^2} \tag{9}$$

[†]It is assumed that the rotors are perfectly flush with the x_B-y_B plane. Therefore, the angular rate of the rotor is only about the z_B-axis. Note that this is a simplification and, in reality, imperfections in the manufacturing and assembly will cause slight deviations. However, the majority of the angular rate will be around the z_B-axis for conventional designs and thus the out-of-axis thrust produced is negligible.

Around hover conditions (i.e. $\alpha_r \approx \frac{\pi}{2}$ and $V \approx 0$), eq. (9) may be rearranged to compute the induced velocity during hover, v_h, through eq. (10) and can be obtained experimentally.

$$v_h = \sqrt{\frac{T}{2\rho A}} \tag{10}$$

Outside of hover conditions, however, the thrust variance effect comes into play [7] and the additional thrust, T_a generated by the rotors can be defined as the actual thrust subtracted by the thrust during hover, T_h, as shown in eq. (11).

$$T_a = T - T_h \tag{11}$$

However, to resolve this additional thrust, the induced velocity, v_{in}, needs to be known. Blade element theory may be used to derive another equation for the thrust relating this induced velocity to the total thrust. Equation (12) represents the thrust derived from blade element theory [27, 29]. It is important to emphasize that eq. (12) is ill-suited for conditions where the quadrotor is close to horizontal surfaces (e.g. roof or ground) since it neglects the ground effect.

$$T = \frac{\rho abc \sum_{j=1}^{4} \omega_j^2 R^3}{2} \left(\frac{\theta_r}{3} + \frac{V^2 \cos^2(\alpha_r)\theta_r}{2\sum_{j=1}^{4} \omega_j^2 R^2} + \frac{V \sin(\alpha_r) + v_{in}}{2\sum_{j=1}^{4} \omega_j R} \right) \tag{12}$$

In eq. (12), a represents the lift curve slope of the rotor, b the number of blades, R denotes the radius of the rotor, c the blade chord, and θ_r the pitch of the rotor blade. In the underlying helicopter literature [29], it is assumed that the chord, c, is constant. This assumption is subsequently carried over into quadrotor literature by Powers et al. [27] who further simplify the blade geometry by assuming fixed pitch blades, contributing to some of the observed modelling errors in [27]. Furthermore, due to their roots in helicopter literature, both eq. (9) and eq. (12) fail to consider interaction effects between rotors and also lead to modelling errors.

In eq. (12), the parameters a, b, c, R, and θ_r are constant for a quadrotor [7]. Therefore, these terms can be lumped into constant coefficients and eq. (12) can be rewritten as:

$$T = \kappa_1 \sum_{j=1}^{4} \omega_j^2 + \kappa_2 V^2 \cos^2(\alpha_r) + \kappa_3 \left(V \sin(\alpha_r) + v_{in}\right) \sum_{j=1}^{4} \omega_j \tag{13}$$

In near hovering conditions, eq. (13) simplifies to eq. (14).

$$T_h = \kappa_1 \sum_{j=1}^{4} \omega_j^2 + \kappa_3 v_h \sum_{j=1}^{4} \omega_j \tag{14}$$

Substituting eq. (13) and eq. (14) into eq. (11) and rearranging for the additional thrust, T_a, yields

$$T_a = \kappa_2 V^2 \cos^2(\alpha_r) + \kappa_3 \left[V \sin(\alpha_r) + (v_{in} - v_h)\right] \sum_{j=1}^{4} \omega_j \tag{15}$$

Therefore, the true induced velocity, v_{in}, can be computed using eq. (16) [2, 28] which can subsequently be used to obtain the thrust variance.

$$v_{in} = \frac{v_h^2}{\sqrt{(V \cos\alpha_r)^2 + (v_{in} - V \sin\alpha_r)^2}} \tag{16}$$

Note, however, the presence of the induced velocity, v_{in}, on both sides of eq. (16). Therefore, for an arbitrary α_r and relative velocity, V, eq. (16) is now a fourth order polynomial with respect to the induced velocity, v_{in}.

Solving the resultant quadratic expression, for hovering conditions, leads to a solution given by eq. (17) [7, 27]. Note that this solution is analogous to the simplified thrust model extensively used in existing literature and therefore highlights the limitations of the simplified model. Namely, that it is only valid in near-hover conditions (i.e. $V \approx 0$ and $\alpha_r \approx \frac{\pi}{2}$).

$$T = \kappa_4 \sum_{j=1}^{4} \omega_j^2 \tag{17}$$

While developing their high-speed models, Sun et al. [7] found that the simple pitching moment model (i.e. eq. (7)) is inadequate for describing the pitching moment during forward flight. It is suspected that the thrust variance effect also induces a moment variance, which is seldom discussed and analyzed in literature.

2. Blade flapping

Another well documented aerodynamic effect, which is significant for the attitude control of the quadrotor, is the phenomena of blade flapping [2, 28, 30, 31]. During translational flight, the advancing blade (i.e. the blade moving towards the direction of motion) sees a higher relative velocity than the retreating blade (i.e. the blade moving away from the direction of motion) [2] stimulating an imbalance in lift across the rotor plane. The net result is the tilting of the resultant thrust vector away from the direction of motion and an induced moment on the drone if the effective rotor plane is offset with respect to gravity [2, 28]. Moreover, the thrust imbalance across the rotor blades also provokes an asymmetry in the induced drag of the rotor (along the $x_B - y_B$ plane) which is otherwise cancelled out during hover (assuming windless conditions). Therefore, due to the rigidity and flapping of the blades, there is also a net horizontal drag force acting on the quadrotor which may be significant for small quadrotors [31]. In addition, as the propellers for most quadrotors are fixed at the rotor hub, the flapping of the propeller blades causes a moment about the rotor hub. All of these additional forces and moments have implications for the attitude control of the quadrotor.

Fortunately, a lumped sum model has been formulated from helicopter literature to (approximately) account for these effects [7, 31]. For general translational motion, a simplified relation describing the flapping angle is given by eq. (18) where β_{flap} and $\beta_{flap}^{\perp}$ are the steady-state flapping angles of the rotor plane to the relative velocity along and perpendicular to the direction of movement respectively [31].

$$\begin{aligned} \beta_{flap} &= -\tfrac{\mu A_{1c}}{1-\frac{1}{2}\mu^2} \\ \beta_{flap}^{\perp} &= -\tfrac{\mu A_{1s}}{1+\frac{1}{2}\mu^2} \end{aligned} \tag{18}$$

In eq. (18), μ denotes the advance ratio which is defined as the ratio of the horizontal velocity of the rotor to the linear velocity of the rotor tip (see eq. (19); R gives the radius of the rotor) while A_{1c} and A_{1s} are positive constants [31]. Note that eq. (18) is derived using the virtual hinge model, which is described and experimentally verified for a quadrotor in [2]. A virtual hinge model is necessary since the flapping angle equations used in traditional helicopter literature assume that the rotor blades are hinged at the rotor hub, which is typically not the case. This assumption results in an over-prediction of the true flapping angle of the rotor plane [2]. Instead, to more accurately model the flapping angle, Hoffman et al. [2] propose that an effective (i.e. virtual) hinge point somewhere along the blade's length be used instead, enabling the continued use of the hinged model developed in helicopter literature. However, as highlighted by Mahony et al. [31], this virtual hinge model results in a phase shift between the sine and cosine components, which is captured in the constants A_{1c} and A_{1s}. In any case, these modifications only emphasize the limitations of adapting models developed in helicopter literature and the lack of (and thus, need for) native models in quadrotor literature.

$$\mu = \frac{\sqrt{V_x^{B^2} + V_y^{B^2}}}{\omega_z^B R} \tag{19}$$

If the stiffness of the rotor is assumed to be similar to a torsional spring, then the induced drag can be modelled as directly proportional to the flapping angle [31]. This approximation is suitable for describing the first bending mode of the rotor and the small flapping angles involved [2]. Note that pioneering literature often assumes a linear relationship between the quadrotor velocity and the induced drag terms. Such a relationship has been shown experimentally to hold for low-speed conditions [2, 30, 31]. However, it is unclear if this relation holds at higher velocities.

As aforementioned, there are two moments which arise due to the flapping of the rotor blades. The first is a moment, $M_{flap_{lon}}$, generated by longitudinal component of the offset thrust, $T_{flap_{lon}}$, due to the tilting of the rotor plane. The second is the moment generated around the motor hub due to the stiffness, κ_{flap}, of the rotor blades. These moments are given by eq. (20) and eq. (21) respectively [2, 28]. Both of these moments are a function of the blade flapping angle, β_{flap}. In eq. (20), l_{cg} denotes the offset between the rotor plane and the quadrotor center of gravity (c.g.).

$$M^B_{flap_{lon}} = T^B_{flap_{lon}} \sin(\beta_{flap}) l_{cg} \tag{20}$$

$$M^B_{flap_{hub}} = \kappa_{flap}\beta_{flap} \tag{21}$$

It is typically assumed that the advance ratio for the quadrotor is small [31] and thus the higher order terms associated with the advance ratio, μ, in eq. (18) are often ignored. However, this assumption only holds if the translational (i.e. body) velocity of the quadrotor is much smaller than the linear velocity at the rotor tips. While this may be suitable for low-speed indoor flight, it is not the case for high-speed outdoor flight. Much remains unknown regarding the effects of blade flapping at higher velocities and during aggressive manoeuvring. Nonetheless, it is appears as though (powers of) the advance ratio and body velocities play an integral role for both the forces and moments experienced by the quadrotor.

D. Employed quadrotor platform: The MetalBeetle

The MetalBeetle, shown in fig. 2, is built in-house and is composed of commercially available parts. This quadrotor is moderately sized with a distance of about 155 mm between the diagonally opposing rotor hubs. BetaFlight is elected as the flight controller for the MetalBeetle. The motivation behind using BetaFlight lies in the fact that outdoor flight data was obtained through manual, piloted, flight and these controllers are open source and well-received by the FPV community. Manual flight is needed to be able to facilitate the high-speed and aggressive flight desired for outdoor model identification in a safe manner. Manual flight also allows for more flexibility in permissible flight conditions and achievable manoeuvres. However, this inherently stimulates variability between flights, challenging the repeatability of manoeuvres and complicating validation process for the models. With a compatible board, another advantage of BetaFlight is its on-board data-logging capabilities. This is especially useful for recording the quadrotor's rotor speeds.

The rotor speeds themselves are measured by the ESC (electronic speed controller). Note that the ESC does not measure the rotor's RPM directly, but rather a proxy of this through the revolutions of the magnetic poles of the rotor hub and is known as the $eRPM$ (electronic RPM). The true RPM may be derived from the $eRPM$ by multiplying the number of poles by the $eRPM$ value. Hence, the $eRPM$ is essentially a scaled down version of the RPM.

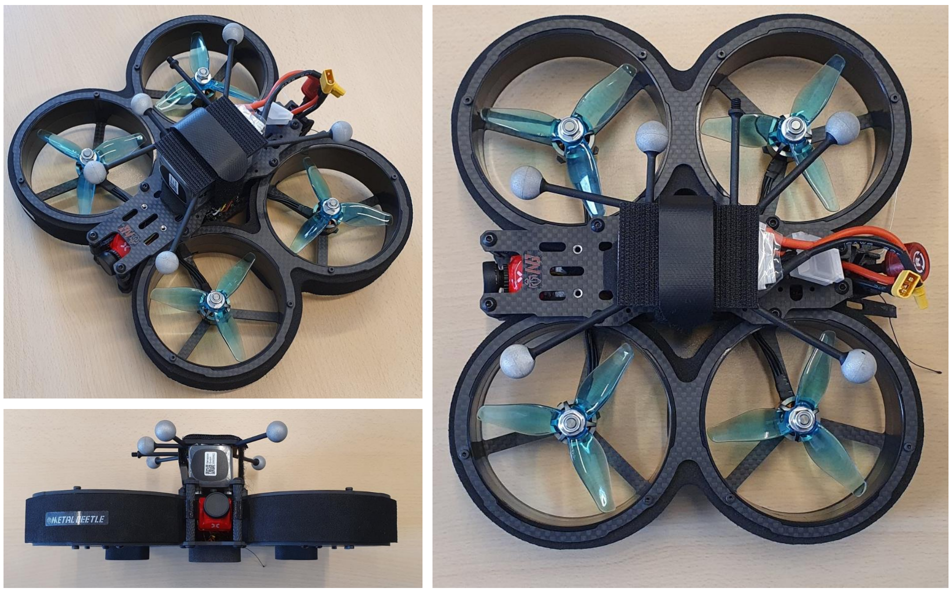


Fig. 2 Top, front and perspective views of the MetalBeetle quadrotor. In these images, the MetalBeetle is shown with the OptiTrack IR marker rig (only used for indoor flights, see section VIII.D) strapped above its battery.

The IMU of the MetalBeetle is an MPU6000 composed of a 3-axis accelerometer and gyroscope. This sensor therefore accounts for the estimation of the MetalBeetle's attitude, body rotational rate, and acceleration. The MPU6000 is commonly used sensor in many FPV quadrotor builds but is now (from 2017 on-wards) been marked NR/ND (i.e. not recommended for new designs) due to the availability of better sensors. For outdoor flights, the MetalBeetle relies on a TBS M8.2 GPS module to obtain velocity state information with a rated refresh rate of between $1-10\ Hz$. This GPS-module also sees frequent use in quadrotor builds[§].

The MetalBeetle is powered by a 4S 850 mAh Tattu LiPO battery. The 4S indicates that there are four battery cells with a capacity of 850 mAh. In general, a higher cell count (i.e. S-count) equips the quadrotor with more power, and thus allows it to fly more aggressively. While different cell counts may be used with the MetalBeetle, all flights conducted for the purposes of this thesis involve only the 4S configuration. The moment of inertia and mass of the MetalBeetle are summarized in table 1. The moment of inertia was obtained through measurements from a Trifilar Pendulum with the propellers removed.

Table 1 Properties of the MetalBeetle

Quadrotor Model	**Mass** $[g]$	**Moment of Inertia** $[kg \cdot m^2]$		
		I_{xx}	I_{yy}	I_{zz}
MetalBeetle	393	9,67E-04	1,02E-03	1,60E-03

III. System identification methods

Following from the analytical models of quadrotor, and additional observations made from flight data, the fundamental states which compose the quadrotor models are first motivated here. Subsequently, a normalization scheme based on that of Sun et al. [8] for these states is proposed, with some modifications made to improve identification performance. Also given in this section is a brief summary of the principles behind the selected system identification techniques: polynomial stepwise regression and artificial neural networks (ANNs). The novel hybrid approach which combines the merits of both techniques is also proposed. For each of the chosen system identification techniques, the input variables (e.g. fixed and candidate regressors for step-wise regression) are also described and motivated to conclude their respective sections. Each of the system identification methods are also accompanied by prediction intervals following the procedure outlined in **Part I: On prediction intervals**. Finally, a concise overview of the system identification pipeline is presented.

A. Constituent states for model identification

Having outlined some fundamental and analytical models of the quadrotor, a state vector encompassing the essential modelling states may be derived. From the first principle force model of the quadrotor (i.e. eq. (3)), it is evident that the body velocities, $\mathbf{V} = [u, v, w]^T$, and the body rotational rates, $\mathbf{\Omega} = [p, q, r]^T$ are essential for describing the fundamental dynamics of the quadrotor.

Literature on thrust variation suggests that the induced velocity, v_{in}, angle of attack of the rotor blades, α_r, and the individual rotor speeds, ω, are also integral components which influence quadrotor's dynamics. While v_{in} can be obtained through eq. (16), the rotor's angle of attack is not available directly from the sensors available on the employed quadrotors. Instead, it may be related to the attitude of the quadrotor, $[\phi, \theta, \psi]^T$, namely, the pitch, θ, and roll, ϕ, of the quadrotor. Therefore, the quadrotor's attitude should be included in the quadrotor models. Note that, while the yaw angle does not directly influence the angle of attack of the rotor blades, it may nonetheless be informative for subsequent models and can be indicative of wind contamination in outdoor flight. Likewise, it may be useful for capturing non-stationary effects related to - for example - the average direction of wind for online applications. However, such an endeavour is out of the scope of this paper but is nevertheless a future objective.

To aid the model selection process, the quadrotor attitudes are represented by their trigonometric identities (i.e. $\sin()$ and $\cos()$) to accommodate their periodic nature. This also bounds, and effectively normalizes, the attitude. Likewise,

[§]Note that indoor flights of the MetalBeetle are also conducted in this research as a comparative model, the results of which may be found in section VIII.D. For these indoor flights, an external motion capturing system, OptiTrack, is used to estimate the MetalBeetle's velocity by monitoring IR markers attached to the quadrotor (see fig. 2).

the individual rotor speeds are represented by the total rotor speed (see eq. (22)) seeing as the measured accelerations are a cumulative effect of the individual rotor speeds. This also simplifies the model structures and mitigates over-fitting.

$$\omega_{tot} = \sum_{i=1}^{4} \omega_i \tag{22}$$

However, the differences in thrust between rotors are nonetheless useful and informative for model identification, especially for the moment models. Therefore, the control moments, $[U_p, U_q, U_r]^T$ (see eqs. (6) to (8)), are included to represent the thrust differentials between the rotors. While these are especially important for the moment models, they may also anticipate forces.

Furthermore, the effects of blade flapping reveal the importance of including the advance ratio terms, $\boldsymbol{\mu} = [\mu_x, \mu_y, \mu_z]^T$, in addition to the velocities. The advance ratio may be defined along each axis as described by eq. (23) [8]. While not explicitly described in the prevailing literature, flight observations and initial modelling revealed that the advance ratio of the induced velocity, $\mu_{v_{in}}$, given by eq. (24), is also an influential parameter for modelling. Consequently, it is also included in the model variable candidates.

$$\mu_x = \frac{u}{\frac{1}{4}\sum_{i=1}^{4} \omega_i R} \quad , \quad \mu_y = \frac{v}{\frac{1}{4}\sum_{i=1}^{4} \omega_i R} \quad , \quad \mu_z = \frac{w}{\frac{1}{4}\sum_{i=1}^{4} \omega_i R} \tag{23}$$

$$\mu_{v_{in}} = \frac{v_{in}}{\frac{1}{4}\sum_{i=1}^{4} \omega_i R} \tag{24}$$

In summary, the fundamental variables used for the identification of the quadrotor models are given by eq. (25).

$$X = [u, v, w, v_{in}, p, q, r, \omega_{tot}, \sin\phi, \cos\phi, \sin\theta, \cos\theta, \sin\psi, \cos\psi, U_p, U_q, U_r, \mu_x, \mu_y, \mu_z, \mu_{v_{in}}] \tag{25}$$

B. Normalization of quadrotor states

It is commonplace in aerospace literature to normalize the states of a system for modelling to facilitate a comparison between different platforms and to ensure that identified models remain valid across different conditions, such as changing air density. Sun et al. [8] propose a novel non-dimensionalization routine for quadrotors, under the assumption that the aerodynamic forces and moments are mainly generated by the rotor system. Indeed, it is difficult to quantify the rotor-local aerodynamic effects since the measured states account for their cumulative effect. Therefore, to represent this combined effect, the average rotor speed is used as the dimensionalizing velocity and is defined in eq. (26), for a given data sample.

$$\bar{\omega} = \sqrt{\frac{\sum_{i=1}^{4} \omega_i}{4}} \tag{26}$$

The individual rotor speeds can then be normalized with respect to this averaged rotor speed through eq. (27). Note that a singularity occurs when all rotors are stopped, which is true when the quadrotors are disarmed. Fortunately, the 'air mode' of BetaFlight maintains an idle thrust while the quadrotor is armed, thereby preventing this singularity. However, the magnitude of this normalized force is disproportionally large for the period where the rotors spin-up to their idle (or hovering) states. Moreover, during some manoeuvres, the rotor speeds occasionally drop below this 'idle' thrust bound resulting in high magnitude normalized terms. Therefore, a lower-bound, $\bar{\omega} \geq \omega_{IDLE}$, is imposed on the the averaged rotor speed where ω_{IDLE} corresponds to the idle (or minimum eRPM¶) rotor speed.

$$\bar{\omega}_i = \frac{\omega_i}{\bar{\omega}} \tag{27}$$

Since the total rotor speed is used as an aggregate state (recall eq. (25)) of the individual rotor speeds, Sun et al. [8] propose a normalization of this entity through the average rotor speed, $\bar{\omega}$. However, this normalization scheme forfeits information pertaining to the current thrust level. As such, the same value of $\bar{\omega}_{tot}$ is obtained for minimum and maximum thrust, and indeed for any case where the thrust level across all rotors is equal. This is undesirable since

¶Recall that the quadrotor ESCs do not measure the rotor speed directly, but rather a proxy in the form of the electronic RPM (eRPM) which defines the number of revolutions of the poles of the rotor hub.

the dynamics between these cases may differ considerably. Instead, here, the total rotor speed is normalized by the maximum rotor speed, ω_{MAX}, as shown in eq. (28).

$$\bar{\omega}_{tot} = \frac{\sum_{i=1}^{4} \omega_i}{\omega_{MAX}} \tag{28}$$

In accordance with the normalization scheme of Sun et al. [8], the normalized variants of the rolling, pitching, and yawing control moments may be computed using the normalized individual rotor speeds through eq. (29), eq. (30), and eq. (31) respectively.

$$\bar{U}_p = (\bar{\omega_1} + \bar{\omega_4}) - (\bar{\omega_2} + \bar{\omega_3}) \tag{29}$$

$$\bar{U}_q = (\bar{\omega_1} + \bar{\omega_2}) - (\bar{\omega_3} + \bar{\omega_4}) \tag{30}$$

$$\bar{U}_r = S_R \left[(\bar{\omega_1} + \bar{\omega_3}) - (\bar{\omega_2} + \bar{\omega_4})\right] \tag{31}$$

The advance ratio is, by definition, scaled by the rotor speed. Hence, the normalized advance ratios along each axis may be defined as

$$\mu_x = \frac{u}{\bar{\omega}R} \quad , \quad \mu_y = \frac{v}{\bar{\omega}R} \quad , \quad \mu_z = \frac{w}{\bar{\omega}R} \tag{32}$$

However, should the body velocities be normalized using the averaged rotor speed, they would effectively mirror the advance ratios. Therefore, the body velocities are instead normalized by their combined magnitude as shown in eq. (33). A singularity occurs while the quadrotor is stationary in space, which includes the hovering condition or a change of direction. To mitigate this, the offending velocities are simply replaced by zero. Practically, this is done through a lowerbound imposed on the normalizing factor such that $max(\sqrt{u^2 + v^2 + w^2}, 0.01)$ is enforced during normalization. Subsequently, the corresponding values for which $\sqrt{u^2 + v^2 + w^2} < 0.01$ are replaced by zero.

$$\bar{u} = \frac{u}{\sqrt{u^2+v^2+w^2}} \quad , \quad \bar{v} = \frac{v}{\sqrt{u^2+v^2+w^2}} \quad , \quad \bar{w} = \frac{w}{\sqrt{u^2+v^2+w^2}} \tag{33}$$

The advance ratio of induced velocity and induced velocity itself may likewise be normalized through eq. (34) and eq. (35) respectively.

$$\mu_{v_{in}} = \frac{v_{in}}{\bar{\omega}R} \tag{34}$$

$$\bar{v}_{in} = \frac{v_{in}}{\sqrt{u^2 + v^2 + w^2}} \tag{35}$$

Following the procedure of Sun et al. [8], the body angular velocities may be normalized through eq. (36).

$$\bar{p} = \frac{pb}{\bar{\omega}R} \quad , \quad \bar{q} = \frac{qb}{\bar{\omega}R} \quad , \quad \bar{r} = \frac{rb}{\bar{\omega}R} \tag{36}$$

Finally, the normalized forces and moments can be obtained through eq. (37) and eq. (38) respectively, where ρ represents the the air density and R the radius of the rotors. In eq. (37), $\mathbf{C}_F = \left[C_x, C_y, C_z\right]^T$ denotes the normalized forces and $\mathbf{F} = \left[F_x, F_y, F_z\right]^T$ gives the dimensional forces along the body axes. Likewise, in eq. (38), $\mathbf{C}_M = [C_l, C_m, C_n]^T$ denotes the normalized moments, $\mathbf{M} = \left[M_x, M_y, M_z\right]^T$ gives the dimensional moments about the body axes, and b relates to the quadrotor's geometry and size. Given the symmetry of most quadrotors (including those used in this paper), the moment arm length along the x_B or y_B axes may be used to represent this quantity. Here, the arm along y_B is taken.

$$\mathbf{C}_F = \frac{\mathbf{F}}{\rho(4\pi R^2)(R\bar{\omega})^2} \tag{37}$$

$$\mathbf{C}_M = \frac{\mathbf{M}}{b\rho(4\pi R^2)(R\bar{\omega})^2} \tag{38}$$

C. Polynomial models for quadrotor system identification

One of the merits of using polynomial based models is that they are simple to apply and interpret [8]. Polynomial models are 'trained' through regression, typically in the form of eq. (39) where $z \in \mathbf{R}^N$ denotes the target measurement data of length N. The matrix A represents arbitrary combinations (e.g. power series) of the independent variables (in this case, the quadrotor states) wherein the first column is a constant vector representing the bias vector. The parameters of the model (i.e. the polynomial coefficients) are denoted by Θ. Therefore, $A\Theta$ represents the predictions of the model. The residuals between these predictions and the measurement data is denoted by ϵ.

$$z = A\Theta + \epsilon \tag{39}$$

For systems which are linear in the parameters, the optimal model parameters, $\hat{\Theta}$, which minimize the error residuals in eq. (39) can be estimated through ordinary least squares (eq. (40)).

$$\hat{\Theta} = (A^T A)^{-1} A^T z \tag{40}$$

It is clear from eq. (39) that the model performance depends on the choice of the regressor matrix, A. The complexity of A is a function of the chosen regressors and the number of thereof. Increasing the complexity of A may lead to lower model residuals at the risk of over-fitting and cost of increased computation load. Conversely, a simple regressor matrix may fail to adequately capture the more nuanced dynamics. This therefore raises the question of how to choose a suitable model structure.

1. The stepwise regression algorithm

Prior knowledge can be applied to determine both the (base) structure and set of candidate regressors of the polynomial, facilitating a so-called 'gray-box' approach to modelling [8]. Regressors from this candidate set may be added to the polynomial model based on their fit with the target data. One such frequently employed in aerospace literature [32, 33] to select these regressors for aircraft models is known as step-wise regression. Sun et al. [7] have successfully used stepwise regression to derive the polynomial model structures from a given set of candidate regressors for the quadrotor platform.

The principle behind this approach is to, in the forward step, select a regressor from the candidate pool which leads to the greatest improvement in model accuracy. Subsequently, in the backward step, the quality of the current selected regressors is reassessed by evaluating the performance of the model with each of the regressors removed. The goal is to identify any regressors which may have become superfluous due to (combinations) of the remaining regressors in the model and remove them. The algorithm terminates once the regressor that was removed in the backward step is the same as that added in the forward step, otherwise the forward step commences again and the process repeats itself.

The stepwise regression is traditionally initialized with eq. (39) and $\mathbf{A} = [1, \ldots, 1]^T$. Note that, if there are 'fixed' regressors (i.e. regressors which will always be included in the polynomial model), these will also form a part of the initial **A** matrix in columns succeeding the bias vector. Candidate regressors are then added to the model based on their correlation with the measured targets, **z**, after orthogonalizing them with respect to the current model regressors in order to capture variations in the measurement data which are otherwise absent in the current model (i.e. absent in **A**) [32]. The adjustment is performed as follows [32]:

1) The candidate regressors in the pool are adjusted to orthogonalize them with respect to the regressors already in **A**. This is accomplished through eq. (41) where ξ_j represents the current regressor.
2) The measurement data, **z**, are likewise modified using eq. (42) to account for the adjustments of the candidate regressors in the previous step. Note that **A** is the same in both eq. (41) and eq. (42)

$$\boldsymbol{\epsilon}_{\xi,j} = \boldsymbol{\xi}_j - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{\xi}_j \tag{41}$$

$$\boldsymbol{\epsilon}_{z,j} = \mathbf{z}_j - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{z}_j \tag{42}$$

Following this adjustment, the correlation of the (adjusted) candidate regressors with the (modified) measurement points can be assessed for each regressor using eq. (43) [32]. The goal is to determine which candidate regressor best explains the remaining variance in the measurements.

$$r_j = \frac{\sum_{i=1}^{N} \left(\epsilon_{\xi,j}[i] - \bar{\epsilon}_{\xi,j}\right)\left(\epsilon_{z,j}[i] - \bar{\epsilon}_{z,j}\right)}{\sqrt{\sum_{i=1}^{N} \left(\epsilon_{\xi,j}[i] - \bar{\epsilon}_{\xi,j}\right)^2 \sum_{i=1}^{N} \left(\epsilon_{z,j}[i] - \bar{\epsilon}_{z,j}\right)^2}} \tag{43}$$

The (unmodified) regressor corresponding to the highest absolute correlation is selected for addition to the model, should this regressor result in a significant improvement to the model itself. The significance of a regressor's contribution is assessed through a F-statistic and is given by eq. (44) whereby F_0 denotes the test statistic and F_{IN} is the user-defined cutoff based on a significance level, α, of the test statistic (i.e. $F_{IN} = F(\alpha; 1, N - p - 1)$) [32]. For aircraft systems, it is often the case that the number of data points is large ($N \gg 100$) and current number of model regressors small ($p < 10$), thus, $N \gg p$ implying that the effect of p on F_{IN} is negligible. Therefore, F_{IN} is assumed to be constant and is commonly set to $F_{IN} = 4$ [32].

$$F_0 = \frac{SS_R(\hat{\mathbf{\Theta}}_{p+j}) - SS_R(\hat{\mathbf{\Theta}}_p)}{s^2} > F_{IN} \tag{44}$$

In eq. (44), $\hat{\mathbf{\Theta}}_p$ gives the model parameters without the added regressor, $\boldsymbol{\xi}_j$, while $\hat{\mathbf{\Theta}}_{p+j}$ represents the model parameters with the addition of $\boldsymbol{\xi}_j$. In both cases, the parameters can be estimated through OLS using eq. (40). $SS_R(\hat{\mathbf{\Theta}})$ denotes the regression sum of squares and is defined in eq. (45) where N gives the number of measurement points and $\bar{z}$ the mean measurement value [32].

$$SS_R(\hat{\mathbf{\Theta}}) = \hat{\mathbf{\Theta}}^T \mathbf{A}^T \mathbf{z} - N\bar{z} \tag{45}$$

s^2 denotes the fit error variance and is defined by eq. (46) with p denoting the current number of regressors in the model and $\hat{y}_i$ the prediction of point i (refer to eq. (47)) [32]. If the condition $F_0 > F_{IN}$ is satisfied, then the regressor $\boldsymbol{\xi}_j$ is added to the model.

$$s^2 \triangleq \hat{\sigma}^2 = \frac{\sum_{i=1}^{N} (z_i - \hat{y}_i)^2}{N - p - 1} \tag{46}$$

$$\hat{\boldsymbol{y}} = \mathbf{A}\hat{\mathbf{\Theta}} \tag{47}$$

Following the addition of a regressor $\boldsymbol{\xi}_j$, all model regressors are examined for redundancy [32]. It is important to note that, during this check, the regressors are not adjusted and therefore remain in their original form. For p model regressors, the quality of each regressor, $\boldsymbol{\xi}_k$, is evaluated through the test statistic defined in eq. (48) where $\hat{\mathbf{\Theta}}_{p-k}$ denotes the parameters corresponding to a model with all p regressors except $\boldsymbol{\xi}_k$ [32].

$$F_{0,k} = \frac{SS_R(\hat{\mathbf{\Theta}}_p) - SS_R(\hat{\mathbf{\Theta}}_{p-k})}{s^2} \tag{48}$$

Should any of the regressors satisfy the condition $F_{0,k} < F_{OUT}$ where F_{OUT} is another user-specified threshold, then regressor with the lowest $F_{0,k}$ is removed from the model. As with F_{IN}, typically $F_{OUT} = 4$ [7, 32]. Should $\boldsymbol{\xi}_k = \boldsymbol{\xi}_j$ (i.e. should the removed regressor be the same as the just added regressor) then the algorithm terminates. If this is not the case, then the algorithm continues adding (and removing redundant) regressors until the termination condition is met.

However, this termination condition does not consider the over-fitting of the model to the measurement data, z [7, 32]. Therefore, additional stopping conditions may be implemented to mitigate this effect. For example, Sun et al. [7] make use of the Predict Square Error (PSE), given by eq. (49), to minimize over-fitting by penalizing the addition of regressors (i.e. promoting fewer model terms) [7, 32]. In eq. (49), $\hat{\mathbf{y}} = \mathbf{A}\hat{\mathbf{\Theta}}$ represents the model predictions and σ^2_{max} is the a constant defined by eq. (50). The second term in eq. (49) represents the penalty for a model with p regressor terms and therefore scales with increasing p.

$$PSE \triangleq \frac{1}{N}(\mathbf{z} - \hat{\mathbf{y}})^T(\mathbf{z} - \hat{\mathbf{y}}) + \sigma^2_{max}\frac{p}{N} \tag{49}$$

$$\sigma^2_{max} = \frac{1}{N}\sum_{i=1}^{N}(z_i - \bar{z})^2 \tag{50}$$

In addition to this stopping condition, an upper limit may be imposed on the number of candidate regressors that may be added to the model. For the purposes of the polynomial models identified here, this is capped at ten candidate regressors (i.e. in addition to any fixed regressors).

2. Polynomial model candidate and structures

Having defined a method for selecting regressors from a candidate pool, the candidate sets themselves need to be constructed. The candidate polynomial models described in this section are loosely based on the high-speed models identified by Sun et al. [8] but are refined through experimentation and observations derived from real flight data.

For brevity, the polynomial structures which house the candidate regressors are expressed in the form $P(x_1, ..., x_k)^n$ where $x_1, ..., x_k$ denote the variables and n the degree of the polynomial. Note that, since the bias term (i.e. constant in the polynomial models) is included by default in the stepwise selection process, it is not included in $P(x_1, ..., x_k)^n$. Furthermore, these polynomials may be multiplied with various constants, typically other (combinations of) variables, to represent a polynomial set. To illustrate this notation, consider the expressions of the polynomial candidates in eq. (51) and their corresponding expanded forms. Here, C_0 denotes the (fixed) bias vector, $P(x, y)^2\{1, a, b\}$ the first polynomial of degree two of variables x, y multiplied with sets: 1 (i.e. one), a, and b with a and b representing some (arbitrary combination) of the variables and/or states. Likewise, $P(x, z)^2\{1, c\}$ denotes a polynomial of degree 2 in x, z multiplied with 1 and the variable c.

$$\begin{aligned}
\hat{Y} = \quad & C_0 + P(x, y)^2 \{1, a, b\} \\
& + P(x, z)^2 \{1, c\} \\
\hat{Y} = \quad & C_0 + 1 \cdot P(x, y)^2 + a \cdot P(x, y)^2 + b \cdot P(x, y)^2 \\
& + 1 \cdot P(x, z)^2 + c \cdot P(x, z)^2 \\
\text{where:} & \\
& P(x, y)^2 = x^2 + 2 \cdot xy + y^2 \\
& P(x, z)^2 = x^2 + 2 \cdot xz + z^2
\end{aligned} \tag{51}$$

The candidate polynomial structure for C_x is given by eq. (52). The first three terms represent the bias, linear velocity along x_B, and the component of the induced velocity along x_B respectively. These terms are considered fixed regressors and thus will always form a part of the polynomial model. The inclusion of the first two terms is inspired by the corresponding F_x model of Sun et al. [8], while the third term was found to be consistently selected by stepwise regression algorithm. It is suspected that this term is related to the component of the thrust variance effect (in particular, the last term in eq. (15)) along the x_B-axis, hence the coupling with the pitch angle. The associated coefficients X_0, X_1, and X_2 are identified when fitting the polynomial model to the training data.

$$\begin{aligned}
\hat{C}_x = \quad & X_0 + X_1\bar{u} + X_2 \sin(\theta) \cdot v_{in}^2 \\
& + P(\bar{u}, |\bar{v}|, \bar{w}, \bar{v}_{in})^4 \{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), \bar{q}, |\bar{r}|, \bar{U}_q, |\bar{U}_r|\} \\
& + P(\mu_x, |\mu_y|, \mu_z, \mu_{v_{in}})^4 \{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), \bar{q}, |\bar{r}|, \bar{U}_q, |\bar{U}_r|\} \\
& + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \{1, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi)\} \\
& + P(\bar{q}, |\bar{r}|)^4 \{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\} \\
& + P(|\bar{U}_p|, \bar{U}_q, |\bar{U}_r|)^4 \{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta)\}
\end{aligned} \tag{52}$$

The first two polynomials (i.e. $P(\bar{u}, |\bar{v}|, \bar{w}, \bar{v}_{in})^4$ and $P(\mu_x, |\mu_y|, \mu_z, \mu_{v_{in}})^4$) in eq. (52) represent the influences of the body velocities, and interactions therein, on F_x. Note that the modulus of the velocity along the y_B-axis, $|\bar{v}|$ and $|\mu_y|$, is used since the effect of v on F_x is expected to be independent of the sign of v. Likewise, F_x is suspected to be independent of the direction of the yaw rate, r, but may be modulated in some way by its magnitude. The extensive polynomial sets - derived from flight observations - are also provided to capture any (linear-like) interaction effects between the body velocities and other states. For instance, the analytical derivations of the thrust models suggest an interaction between the body velocities - particularly w and v_{in} - and attitude angles. Given the considerable attitudes that the quadrotor may operate at, especially for aggressive manoeuvres, the relevant attitude angles are likely integral to the quadrotor model.

These extreme attitudes also motivate the selection of the third polynomial, $P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4$, which attempts to capture the interaction effects between the rotor speeds and vertical velocities of the quadrotor in the context of these attitude angles. It is expected that the interaction between the pitch angle, θ, and the rotational speed of the rotors, $\bar{\omega}_{tot}$, is especially important. Moreover, as derived in the analytical models of section II.C, the thrust is directly related to the $\bar{\omega}_{tot}$, the attitude, and the interaction of the two.

Any higher order effects of the body rotational rates on F_x, and in relation to the total rotor speed and vertical velocities, are to be captured by the forth polynomial in eq. (52). These terms may forecast changes in force as they constitute the derivatives of the quadrotor's attitude. Similarly, the final polynomial is aimed at forecasting the force based on the differential thrust produced by the rotors, especially since this change in force does not act instantaneously due to, for example, actuator delays and rate limits.

$$\begin{aligned}\hat{C}_y = \; & Y_0 + Y_1\bar{v} + Y_2 \sin(\phi) \cdot v_{in}^2 \\ & + P\left(|\bar{u}|, \bar{v}, \bar{w}, \bar{v}_{in}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi), \bar{p}, |\bar{r}|, \bar{U}_p, |\bar{U}_r|\right\} \\ & + P\left(|\mu_x|, \mu_y, \mu_z, \mu_{v_{in}}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi), \bar{p}, |\bar{r}|, \bar{U}_p, |\bar{U}_r|\right\} \\ & + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \left\{1, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi)\right\} \\ & + P\left(\bar{p}, |\bar{r}|\right)^4 \left\{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\right\} \\ & + P(\bar{U}_p, |\bar{U}_q|, |\bar{U}_r|)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi)\right\}\end{aligned} \tag{53}$$

Equation (53) represents the chosen candidate polynomial structure for the (normalized) force along the y_B-axis, C_y. The motivations for this structure are synonymous to those for C_x, substituting the axis specific differences where relevant (e.g. roll angle, ϕ, instead of pitch, θ).

Many of the motivations behind the selection of candidate regressors for C_z, outlined in eq. (54), are analogous to those behind the candidates of C_x and C_y. However, there are a few additions and differences made to the candidate regressors of C_z. The first five terms denote the fixed regressors wherein $\left(\bar{u}^2 + \bar{v}^2\right)$ and $(\bar{v}_{in} - \bar{w})^2$ follow directly from the analytical models of thrust [8] (recall the thrust variance equations of section II.C). Further note, that the term $(\bar{v}_{in} - \bar{w})^2$ parallels the role of the fixed regressors associated with v_{in} for the C_x and C_y models (i.e. $\sin(\theta)\bar{v}_{in}^2$ and $\sin(\phi)\bar{v}_{in}^2$ respectively). The final fixed regressor, ω_{tot}, directly describes the relation between the total rotor speed and thrust produced and is equivalent to the simple thrust model (i.e. eq. (5)).

$$\begin{aligned}\hat{C}_z = \; & Z_0 + Z_1\bar{w} + Z_2(\bar{u}^2 + \bar{v}^2) + Z_3(\bar{v}_{in} - \bar{w})^2 + Z_4\omega_{tot}^2 \\ & + P\left(|\bar{u}|, |\bar{v}|, \bar{w}, \bar{v}_{in}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), |\bar{p}|, |\bar{q}|, |\bar{r}|, |\bar{U}_p|, |\bar{U}_q|, |\bar{U}_r|\right\} \\ & + P\left(|\mu_x|, |\mu_y|, \mu_z, \mu_{v_{in}}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), |\bar{p}|, |\bar{q}|, |\bar{r}|, |\bar{U}_p|, |\bar{U}_q|, |\bar{U}_r|\right\} \\ & + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \left\{1, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi)\right\} \\ & + P\left(|\bar{p}|, |\bar{q}|, |\bar{r}|\right)^4 \left\{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\right\} \\ & + P(|\bar{U}_p|, |\bar{U}_q|, |\bar{U}_r|)^4 \left\{1, \bar{\omega}_{tot}, \sin(\psi), \cos(\psi)\right\}\end{aligned} \tag{54}$$

Furthermore, eq. (54) includes the interaction effects between the velocities (and, equivalently, advance ratios) and all the attitude angles, rotational rates, and control moments. The motivation for this is that performing an attitude change typically induces a change in thrust across the rotors. The absolute values of the rates and control moments are taken since the direction of change is expected to have negligible influence on the produced thrust.

The candidate polynomial structure for the normalized rolling moment, C_L, is summarized in eq. (55) wherein the first three terms denote the fixed regressors. Both the roll rate and control rolling moment are elected as fixed regressors as they follow directly from the analytical models of the quadrotor and are thus considered fundamental components of the model.

$$\begin{aligned}\hat{C}_L = \; & L_0 + L_1\bar{p} + L_2\bar{U}_p \\ & + P\left(|\bar{u}|, \bar{v}, \bar{w}, \bar{v}_{in}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi), \bar{p}, |\bar{r}|, \bar{U}_p, |\bar{U}_r|\right\} \\ & + P\left(|\mu_x|, \mu_y, \mu_z, \mu_{v_{in}}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi), \bar{p}, |\bar{r}|, \bar{U}_p, |\bar{U}_r|\right\} \\ & + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \left\{1, \sin(\phi), \cos(\phi), \sin(\psi), \cos(\psi)\right\} \\ & + P\left(\bar{p}, |\bar{q}|, |\bar{r}|\right)^4 \left\{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\right\} \\ & + P(\bar{U}_p, |\bar{U}_q|, |\bar{U}_r|)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi)\right\}\end{aligned} \tag{55}$$

The first and second polynomials in eq. (55) are directed at capturing any effects that the body velocities (and equivalently, advance ratios) have on the rolling moment. In particular, due to the effects of blade flapping (recall section II.C.2), it is expected that the regressors associated with the advance ratios constitute relevant elements of the moment models. As some of the moments induced by blade flapping are related to the thrust vector, it is natural to include interaction effects between the velocities and the total rotor speed, ω_{tot}. The other interaction effects are included to mirror the polynomial sets of the force models, again to capture effects of the thrust on the rolling moment. It is assumed that the quadrotor's geometry remains unchanged for the duration of the flight, such that the relevant moment arms are constant. Additionally, the velocities perpendicular to the rotation axis retain their direction in order to capture moment variations arising from the rotor wakes. Consider, for example, a quadrotor flying with a certain velocity, v, and roll angle, ϕ. Due to the wake of the rotors, the rolling dynamics behind rolling forward (i.e. away from the wake) are different than those rolling backwards (i.e. into the wake) as the efficiency of the rotors depends on the local (relative) airspeed. Such effects are observed in the high speed flight experiments of Sun et al. [7, 8, 12].

Indeed these rotor wake effects are important, and often overlooked in quadrotor literature. Hence, the third polynomial also seeks to capture some of the rotor wake effects through the induced velocity terms. It is hoped that the inclusion of the induced velocity can anticipate these effects, and any other emergent effects related to these induced velocities and rotor speeds.

The final two polynomials in eq. (55) relate to any higher order rotational rate and control moment effects, respectively, which may be present during high speed flight. These polynomials are founded in the simple analytical models of the quadrotor, but are aimed at capturing more non-linear effects.

Equation (56) presents the candidate polynomial structure for the (normalized) pitching moment, C_M, constructed with the same underlying motivations as the (normalized) rolling moment model.

$$\begin{aligned}\hat{C}_M = \; & M_0 + M_1\bar{q} + M_2\bar{U}_q \\ & + P\left(\bar{u}, |\bar{v}|, \bar{w}, \bar{v}_{in}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), \bar{q}, |\bar{r}|, \bar{U}_q, |\bar{U}_r|\right\} \\ & + P\left(\mu_x, |\mu_y|, \mu_z, \mu_{v_{in}}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), \bar{q}, |\bar{r}|, \bar{U}_q, |\bar{U}_r|\right\} \\ & + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \left\{1, \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi)\right\} \\ & + P\left(|\bar{p}|, \bar{q}, |\bar{r}|\right)^4 \left\{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\right\} \\ & + P(|\bar{U}_p|, \bar{U}_q, |\bar{U}_r|)^4 \left\{1, \bar{\omega}_{tot}, \sin(\theta), \cos(\theta)\right\}\end{aligned} \tag{56}$$

The candidate polynomial structure for the (normalized) yawing moment model is presented in eq. (57). The motivations behind the selection of the candidate polynomials mirror those of the rolling and pitching moment models, with the exception of one key difference. Unlike the former moment models, wherein the perpendicular velocities maintained their direction, here only the velocity along z_B (i.e. in-line with the yawing axis) retains its direction. This is because, as the quadrotor flies up, it moves away from the rotor wake whereas when it flies down, it goes into its wake. Therefore, the efficiency of the rotors (and thus the control moments) depends on the direction of w. This is not the case for the interaction between the the other velocities and the yawing moment.

$$\begin{aligned}\hat{C}_N = \; & N_0 + N_1\bar{r} + N_2\bar{U}_r \\ & + P\left(|\bar{u}|, |\bar{v}|, \bar{w}, \bar{v}_{in}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), |\bar{p}|, |\bar{q}|, |\bar{r}|, |\bar{U}_p|, |\bar{U}_q|, |\bar{U}_r|\right\} \\ & + P\left(|\mu_x|, |\mu_y|, \mu_z, \mu_{v_{in}}\right)^4 \left\{1, \bar{\omega}_{tot}, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi), |\bar{p}|, |\bar{q}|, |\bar{r}|, |\bar{U}_p|, |\bar{U}_q|, |\bar{U}_r|\right\} \\ & + P(\bar{v}_{in}, \mu_{v_{in}}, \bar{\omega}_{tot})^4 \left\{1, \sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta), \sin(\psi), \cos(\psi)\right\} \\ & + P\left(|\bar{p}|, |\bar{q}|, \bar{r}\right)^4 \left\{1, \bar{\omega}_{tot}, \bar{v}_{in}, \mu_{v_{in}}\right\} \\ & + P(|\bar{U}_p|, |\bar{U}_q|, \bar{U}_r)^4 \left\{1, \bar{\omega}_{tot}, \sin(\psi), \cos(\psi)\right\}\end{aligned} \tag{57}$$

3. Polynomial model prediction intervals

For polynomial models, the prediction interval associated with a given prediction, $\hat{y}_i$, may be obtained through eq. (58) where $t_{N-2}^{1-\alpha/2}$ gives two sided test statistic for the chosen confidence level, $1-\alpha$, and n gives the number of observations. Typically, $n = 1$ when making (single) predictions.

$$\hat{y}_i \pm t_{N-2}^{1-\alpha/2} \frac{\sigma_i}{\sqrt{n}} \tag{58}$$

In eq. (58), σ_i denotes the variance associated with making predictions and may be estimated through eq. (59). In eq. (59), $\mathbf{x_i}$ denotes the input data for which predictions should be made, $\mathbf{A}$ represents the regressor matrix used for training the model, $\mathbf{I}$ gives the identity matrix and σ_e^2 may be approximated through eq. (60).

$$\hat{\sigma}_i^2 = \sigma_e^2 \left(\mathbf{I} + \mathbf{x}_i \left(\mathbf{A}^T\mathbf{A}\right)^{-1} \mathbf{x}_i^T\right) \tag{59}$$

$$\hat{\sigma}_e^2 = \frac{1}{N-2}\sum_{i=1}^{N}\hat{e}_i^2 \tag{60}$$

In eq. (60), N represents the number of samples used for training and $\hat{e}_i = y_i - \hat{y}_i$ denotes the residual error.

D. Artificial Neural Networks for quadrotor system identification

The incomplete and irregular knowledge on the quadrotor models limits, to a certain extent, the use of gray-box models. Therefore, when investigating uncharted regions of the flight envelope, black-box system identification approaches are often employed. Artificial Neural Networks (ANNs) have emerged in recent decades as a particularly popular choice for such black-box modelling due to advancements in computational efficiency [34] and improvements to ANNs themselves, such as the development of unsupervised trainable deep ANNs (DNNs) [35].

ANNs are capable of capturing unknown non-linearities [17, 18] in a system and are thought to be universal function approximators [14]. The latter property is especially powerful since it facilitates the modelling of unobserved states from the measurement data, making ANNs particularly attractive as a system identification technique. It is no surprise, then, that ANNs have successfully been applied to identify models of aircraft [15, 36, 37], rotorcraft [38, 39], and, more recently, quadrotors [11, 14, 16, 40, 41].

1. Principles behind the Artificial Neural Network

An ANN is a system of interconnected artificial neurons designed to emulate biological neural networks. Let x_i denote an input to the neuron with $i \in [1, 2, \ldots, N]$ for N inputs. Each input is subsequently scaled by a corresponding adaptive weight, w_i. Included in the model of the neuron is a bias term denoted by w_0. The purpose of this bias term is analogous to the role of a constant in a polynomial function. The sum of these weighted inputs and bias term, z (see eq. (61)), is subsequently transformed by some activation function, $\sigma(z)$, culminating in the neuron output, v.

$$z = w_0 + \sum_{i=1}^{N} w_i x_i \tag{61}$$

The structured organization of neurons, with the same or different activation functions, defines the network. The most simple NN architecture is perhaps the feed-forward neural network (FNN), illustrated in fig. 3, wherein information only flows from n inputs to m outputs. A general feed-forward neural network is composed of an input layer, K hidden layers, and an output layer. It should be noted that the input layer does not contain neurons and instead relays the inputs to the neurons of the first hidden layer.

Another commonly used NN architecture is the recurrent neural network (RNN), which exploits the temporal invariance of the system to reduce the number of parameters (namely, weights) in the network. As such, RNNs are specialized for time invariant systems and are particularly alluring for system identification and control due to the stationary nature of many systems. Mohajerin et al. [16] successfully applied an RNN in a hybrid configuration with first principle techniques to identify a model of a quadrotor. However, RNNs are less equipped to handle non-stationary effects, which may be ubiquitous during outdoor flight. Therefore, to be able to capture (any) non-stationary effects which occur during high-speed and aggressive outdoor flight while keeping the network structure simple, a FNN architecture is elected for model identification. However, a FNN may be more prone to over-fitting.

The structure of a FNN is determined by the number hidden layers, K, and the number of neurons in a hidden layer, n_k. These variables considerably impact the resultant performance of the ANN. Adding neurons increases the susceptibility of the feed-forward ANN to over-fitting. Such 'wide' ANNs are adept at memorization and begin to model the noise in the data instead of the dynamics behind the data. More neurons also entails more trainable network parameters, demanding more computational effort. However, too few neurons may result in under-fitting whereby the FNN model fails to adequately capture the system dynamics.

An alternative to manipulating the number of neurons is to instead change the number of hidden layers. In general, increasing the number of hidden layers improves the abstraction capabilities of the ANN. However, even single hidden layer feed-forward ANNs are suitable for modelling applications [42]. Unlike increasing the number of neurons, adding more hidden layers facilitates generalization through feature extraction instead of memorization [34]. This property of 'deep' FNNs explains their ever-growing popularity and preference over wide ANNs. This is not to say, however,

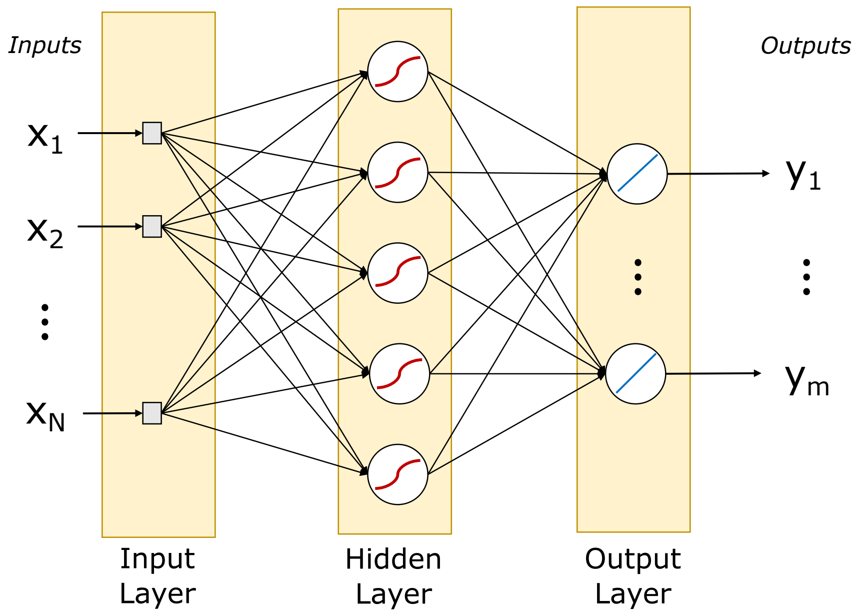


Fig. 3 Network architecture of a single hidden layer feed-forward Artificial Neural Network. Neurons are indicated through circular blocks while the input pass-through nodes are given by square blocks.

that adding layers resolves the issue of over-fitting. In fact, over-fitting is still a considerable issue for deep FNNs. Additionally, as with wide ANNs, the computational demand of training scales with the number of hidden layers.

For ANNs, training is accomplished through a learning procedure known as 'backpropagation' [43–46] which aims to find a set of network weights that minimize the error between the network outputs and desired outputs, for all (especially unseen) inputs. Backpropagation exploits the error between the network output(s) and its target value(s) to update the network weights by traversing the network from outputs to inputs.

$$\frac{\partial J}{\partial w_{ij}^{k}} = \sigma'(z_j^k) v_i^{k-1} \sum_{l=1}^{r^{k+1}} q_l^{k+1} w_{lj}^{k+1} \tag{62}$$

The backpropagation formula for an arbitrary neuron in the network can be found through eq. (62). J represents the cost function being minimized (typically the mean squared error for regression), w_{ij}^k denotes the weight from neuron i in layer $k-1$ to neuron j in layer k. Likewise, w_{lj}^{k+1} denotes the weight from neuron l in layer $k+1$ to neuron j in layer k. σ gives the activation function, z_j^k the weighted sum of inputs into neuron j, and v_i^{k-1} the output of neuron i in layer $k-1$. q_l^{k+1} is given by eq. (63) and describes the change in loss function with respect to the weighted sum of the outputs of the subsequent network layer.

$$q_l^{k+1} = \frac{\partial J}{\partial z_l^{k+1}} \tag{63}$$

Since the gradient gives an indication of the direction in which the weights should be changed, eq. (64) may be used to update the weights proportionally to the gradient through a constant, α [43, 46]. The weights are typically initialized randomly, ensuring that each weight in a given layer is unique[‖].

$$\mathbf{w}_{new} = \mathbf{w}_{old} - \alpha \nabla J(\mathbf{w}_{old}) \tag{64}$$

Where $\nabla J(\mathbf{w}_{old})$ denotes the gradient (expanded in eq. (65)) of all the weights, $w_r \in [1, R]$, in the network.

$$\nabla J(\mathbf{w}_{old}) = \frac{\partial J}{\partial \mathbf{w}_{old}} = \left[\frac{\partial J}{\partial w_{old,1}}, \frac{\partial J}{\partial w_{old,2}}, \dots, \frac{\partial J}{\partial w_{old,R}}\right]^T \tag{65}$$

[‖]Neurons initialised with the same weight will learn the same feature. Hence, for suitable learning, weights should be distinct.

For a database of size, N, the gradient may be computed as

$$\nabla J(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\partial J(\mathbf{t}_n, \mathbf{x}_n, \mathbf{w})}{\partial \mathbf{w}}\right) \tag{66}$$

Taking eq. (66) with the weight update rule (eq. (64)) results in the so-called gradient descent method. For large N, using the gradient descent may be inefficient. Therefore, the database can be split into batches of size, $n << N$, containing data points (i.e. $t_i; x_i$) that are independent of each other and identically distributed. This is known as the stochastic gradient descent, where the gradient may be estimated by eq. (67).

$$\hat{\nabla} J(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial J(\mathbf{t}_i, \mathbf{x}_i, \mathbf{w})}{\partial \mathbf{w}}\right) \tag{67}$$

For some applications, the simple (stochastic) gradient descent may be inefficient (e.g. too slow). In light of this, some extensions to this method are employed in literature to improve the convergence performance. For instance, the velocity of the gradient descent may be used to allow the algorithm to take larger update steps if the gradient is consistently along the same direction. This is known as stochastic gradient descent with momentum, and is defined in eq. (68) where i denotes current update iteration and $\gamma \in [0, 1]$ is a constant representing how influential previous weight updates are [46].

$$\begin{gathered}\Delta\mathbf{w}_{new}[i] = -\alpha\nabla J(\mathbf{w}_{old}[i]) + \gamma\Delta\mathbf{w}_{new}[i-1] \\ \mathbf{w}_{new}[i] = \mathbf{w}_{old}[i] - \Delta\mathbf{w}_{new}[i]\end{gathered} \tag{68}$$

However, too large of a step can result in poor convergence to the minima of the cost function. For instance, the algorithm may be unable to venture into a valley since the step size is wider than this valley. Hence, other methods, such as RMSProp [47], aim to reduce the update step size over time to mitigate this. Better yet, the principles of momentum and decreasing the step size over time can be combined. One such method is ADAM proposed by Kingma and Ba [47]. The choice of the optimizer depends on the requirements imposed on the network (e.g. convergence time and accuracy).

Recall that the goal of the training is to minimize the error between the model predictions and the underlying dynamics of the process in question. The cost function used in backpropagation actually measures the empirical loss of the model, which is the error to the *training* data. Hence, a good empirical loss does not necessarily reflect good overall model performance, since the ANN may be capturing noise and artefacts in the training data. This is mitigated through regularization which seeks to improve the performance of an artificial neural network for states outside of the training data set.

One regularization strategy is to exploit the stochasticity of training process. Due to the random weight initializations, emergent ANN models are typically different from one initialization to the next, culminating in different predictions across these ANNs for the same inputs[**]. Therefore, multiple ANNs can be trained on the same data set and their predictions combined, in some way, to give more consistent overall model predictions [48]. This can be achieved by simply averaging the ensemble outputs. Such an model ensemble approach has seen success in ANN literature (see, for instance, the improved ensemble performance of Krizhevsky et al. [44]).

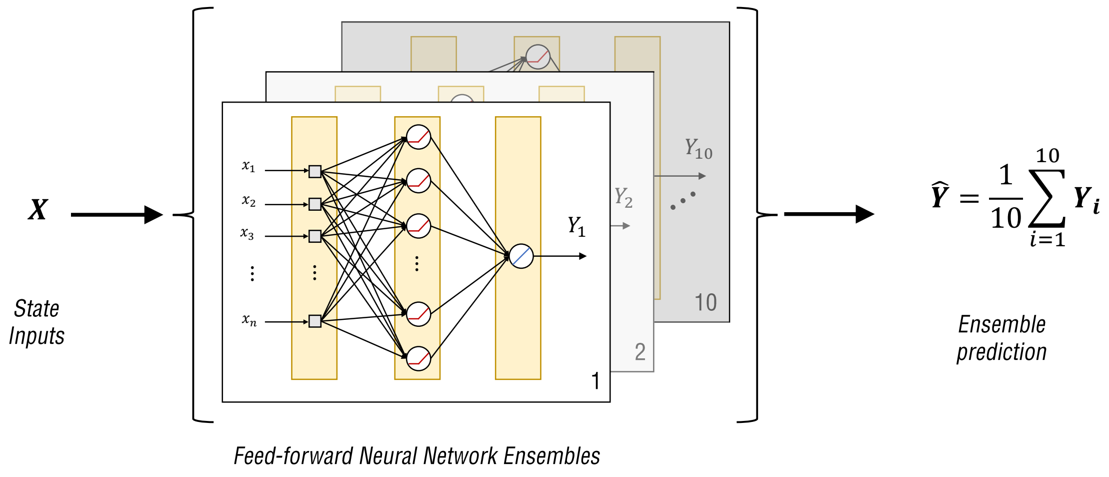
Training a single model can be time consuming and resource intensive, thus training model ensembles may not be practical depending on the application. A more practical implementation of this concept is to use 'dropout' while training [44]. For every forward pass of the backpropagation algorithm, neurons are randomly 'dropped out' by setting their weight to zero with a certain probability. This alters the network architecture for each pass, therefore acting as a 'different' model. Since neurons can dropout, the co-dependence of neurons is reduced and they are subsequently encouraged to learn more general features [44]. Once the training is complete, all neurons are included in the model, scaled by the probability of dropping out [44].

Both these regularization techniques will be applied to the identified ANN models in this paper to mitigate over-fitting. Moreover, each ANN in the ensemble will be trained on a different subset of the training data.

2. *Proposed neural network architecture for model identification*

To keep network structures simple, the ANN-based force and moment models are composed of a collection (i.e. ensemble) of ten FNN models, with each FNN composed of an input layer, a single hidden layer containing 50 neurons,

[**]This is somewhat similar to the variation in polynomial coefficients for the same model structure, but trained on a different subset of the training data.



X
State Inputs
x_1
x_2
x_3
x_n
Y_1
Y_2
Y_{10}
1
2
10
Feed-forward Neural Network Ensembles
$\hat{Y} = \frac{1}{10}\sum_{i=1}^{10} Y_i$
Ensemble prediction

Fig. 4 Illustration of the ensembled Feed-forward Neural Network architecture employed for model identification.

and an output layer. To reiterate, an FNN structure was elected over an RNN structure to be able to capture any non-stationary effects present in the outdoor flight data and to keep the network structures simple. An illustration of this network architecture is provided in fig. 4.

The inputs for the FNNs are further normalized (i.e. in addition to the normalization discussed in section III.B) such that each of the inputs roughly is zero-mean with a standard deviation of one. This is done to improve the learning characteristics and to ensure that each input state is represented equally in magnitude. This also facilitates a comparison between the final input weight magnitudes to provide insight into which states the FNNs consider important [19].

The number of neurons in the hidden layer were constrained to be close in magnitude to the input vector (i.e. the state vector, eq. (25)) to mitigate over-fitting, while numerous enough to capture the model complexities. From preliminary results, 50 neurons appeared to produce desirable results in terms of model accuracy without apparent over-fitting. The FNNs are also compiled with a neuron dropout rate of 0.1 to promote regularization.

The rectified linear unit (ReLU) is chosen as the hidden layer activation function given its simplicity and success in ANN applications [45]. Indeed, while the ReLU function suffers from the dying ReLU problem (i.e. where certain neurons stop learning due to a zero gradient for negative inputs), the more robust ReLU derivatives (e.g. leaky ReLU) did not seem to result in significant model improvements during prelimenary testing. Consequently, the ReLU activation function is retained as the hidden activation function for its simplicity. As the modelling problem is effectively one of regression, a linear activation function is selected for the output layer. Furthermore, ADAM is chosen as the optimizer function due to its capabilities [47].

$$QD = MPIW_C + N\lambda \max\left(0, (1-\alpha) - PICP_s\right)^2 \tag{69}$$

From the results of **Part I: On prediction intervals**, the quality-driven direct PI method is elected for the ANN models since the errors involved with predictions may not be normally distributed and the level of noise is not found to heavily contaminate the dynamics of interest. Subsequently, a custom loss function given by eq. (69) is used. Here, N denotes the number of samples, λ is a hyper-parameter, and $(1-\alpha)$ represents the chosen confidence level. $MPIW_C$ denotes the mean width of the estimated prediction intervals (PIs), given that the target sample is contained within the estimated interval, normalized by the range of the target data as shown in eq. (70). In eq. (70), N_C describes the total number of target samples contained within the prediction interval with upper bound U and lower bound L.

$$MIPW_C = \frac{1}{N_C}\sum_{i=1}^{N_C}(U_i - L_i) \tag{70}$$

$$PICP = \tfrac{1}{N}\textstyle\sum_{i=1}^{N} c_i$$

where

$$c_i = \begin{cases} 1, & \hat{y}_i \in [L_i, U_i] \\ 0, & \text{otherwise} \end{cases} \tag{71}$$

The $PICP$ represents the coverage probability of the estimated PIs and counts the proportion of the target data that is contained within the estimated PIs and is described in eq. (71) where N denotes the total number of samples and y_i the true target value. In eq. (69) the 'soft' variant of the $PICP$ is used and summarized in eq. (72) to improve convergence characteristics where $\sigma(\cdot)$ denotes the sigmoid function and s is some softening factor.

$$PICP_s = \frac{1}{N}\sum_{i=1}^{N}\sigma(s(y_i - \hat{y}_{L,i}))\cdot\sigma(s(\hat{y}_{i,U} - y_i)) \tag{72}$$

Regarding the training parameters, 150 epochs are used for training with data batch sizes of 400. These hyper-parameters were tuned in the early, exploratory, stages of model identification and remain fixed across all the FNN models. Moreover, to encourage variation among the ensemble models, each model is initialized on a different set of random weights and trained on a different subset of the training data set. All of the FNNs are constructed using the `TensorFlow` and `keras` libraries in Python. Moreover, to be directly comparable with the individual polynomial models, ensembles of FNNs of the aforesaid structure are created for each of the forces and moments (e.g. one ANN ensemble of 10 FNNs for F_x alone). The aggregate model prediction is taken as the average of the ensemble predictions.

E. Novel hybrid approach for quadrotor system identification

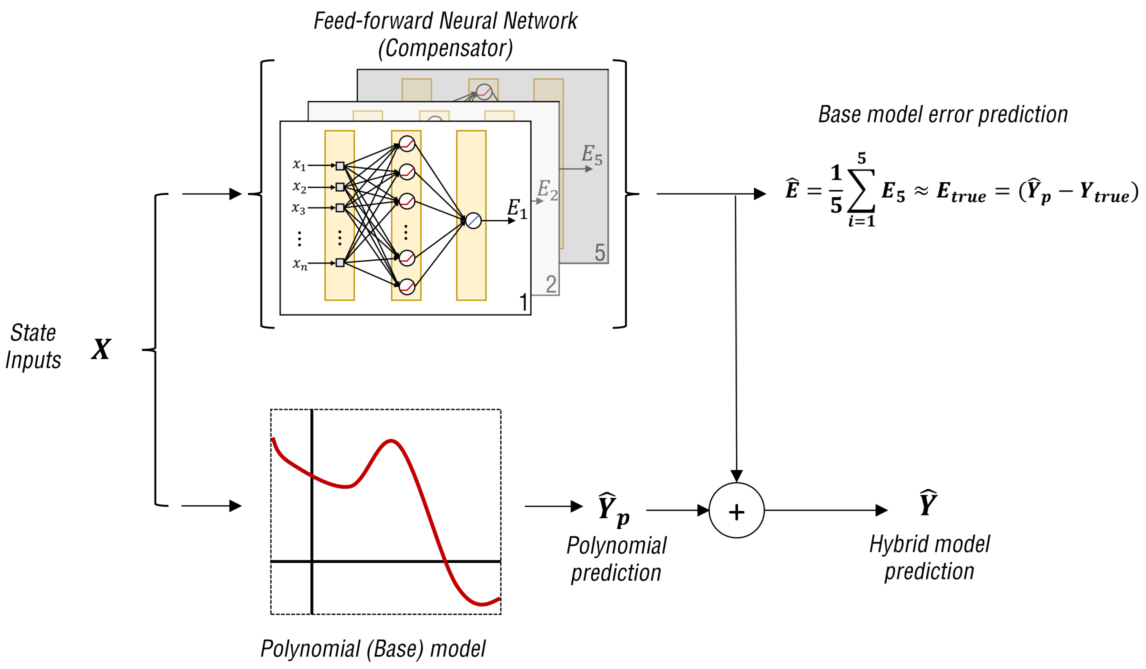
The principle behind this hybrid approach is to leverage the transparency and interpretability of the polynomial models with the non-linear approximation power of ANNs. Therefore, the polynomial models are targeted at approximating the stationary dynamics of the quadrotor while the ANN components accommodate the more complex, or otherwise non-stationary, effects. Since the ANN model is aimed at reducing the modelling errors of the underlying polynomial model, it 'compensates' for the modelling deficiencies of the polynomial model. As such, the ANN compensator is trained to predict the residual errors between the polynomial model predictions and the target values, for the associated input (i.e. state) vector. These ANN predicted residual errors are subsequently removed from the polynomial model predictions to improve estimates of the targets.

This hybrid approach may also be envisioned in an adaptive modelling framework, whereby the polynomial component acts as a base, fixed, model identified offline while the ANN is trained during flight. This ensures that the quadrotor is able to fly safely by initially relying solely on the polynomial model, while facilitating the improvement of the model over time by means of the ANN. Once the ANN model augmentation is reliable enough (for instance, through acceptable reliability metrics as discussed in **Part I: On prediction intervals**), it may be introduced into the system. This delayed transition from polynomial-only to the full hybrid model adds an extra layer of safety, ensuring that the predictions of the ANN are trustworthy enough before depending on them.

1. Hybrid model structure

Since no modifications are made to the polynomials directly, the polynomial structures are identical to those described previously in section III.C.2 (i.e. eqs. (52) to (57)). In fact, the hybrid models only add an ANN component to the polynomial models identified through step-wise regression and may therefore be seen as an extension of the polynomial models.

Likewise, the ANN compensator utilizes the same structure as the ANN-only models with the exception of fewer ensembles and neurons. The number of ensembles is lowered from ten to five for a reduced computational load during training to accommodate the envisioned role of the ANN compensator in the hybrid approach; that is, that the ANN compensator targets the non-stationary effects prevalent during outdoor flight in an online capacity. Moreover, five ensembles have been shown to be sufficient capturing inter-model variations [22, 49] while mitigating over-fitting.



Feed-forward Neural Network (Compensator)
Base model error prediction
$\hat{E} = \frac{1}{5}\sum_{i=1}^{5} E_5 \approx E_{true} = (\hat{Y}_p - Y_{true})$
E_1
E_2
E_5
State Inputs
X
$\hat{Y}_p$
Polynomial prediction
$\hat{Y}$
Hybrid model prediction
Polynomial (Base) model

Fig. 5 Illustration of the hybrid model structure composed of a polynomial component and a feed-forward neural network compensator.

Related to the issue of over-fitting, the number of neurons are further reduced from 50 to 30. As the ANN compensator aimed at predicting the polynomial modelling errors, it is more susceptible to noise and thus prone to over-fitting. This is especially true if the underlying polynomial already produces a good fit. It was found during the exploratory phase of model identification, that increasing the number of neurons beyond 30 occasionally lead to obvious over-fitting. An overview of the hybrid model structure is illustrated in fig. 5.

As the hybrid models described in this paper are unique to this paper, there are no established methods in literature to construct PIs for the hybrid models. Instead, a composite form of the PI estimation techniques of the polynomial and ANN approaches may be formulated to derive the hybrid PIs. Perhaps the most straightforward approach is to simply sum the model prediction error variances obtained from the hybrid model constituents, as shown in eq. (73), for which the interval bounds can be obtained through eq. (58). This is compatible with the envisioned modularity of the hybrid approach, wherein the polynomial model provides the predictions, and associated PIs, while the ANN-compensator is training. However, it is important to highlight that the addition of the two component PIs inherently culminates in wider and more conservative PI estimates.

$$\hat{\sigma}_{HYBRID} = \hat{\sigma}_{POLY} + \hat{\sigma}_{ANN} \tag{73}$$

F. Quadrotor model identification pipeline

In order to expedite the model identification process, a modular system identification pipeline is developed in Python which incorporates the aforementioned identification methods. The modular design of the pipeline allows for the addition of other system identification techniques. A cohesive `Model` class is developed to standardize the identification, predictions, evaluations, and storage of the models.

The desired system identification technique must be specified upon the initialization of the `Model` object. Models are then prepared for training through the `Model.compile` decorator method. Due to the inherent differences between various system identification techniques, this method is tailored to the chosen technique upon initialization. The `Model.compile` method prepares and standardizes the models such that technique-specific terms do not need to be specified in subsequent methods but are nonetheless configurable by the user, if desired. Models may subsequently be identified through the `Model.train` method and predictions made through the `Model.predict` method. The validity of the identified models may be checked through the `Model.evaluate` method. Furthermore, the status of models at any stage may be inspected through the `Model.summary` method.

The pipeline takes a `pandas DataFrame` object containing the training data as inputs with the samples along the rows and system states along the columns. Note that the `Model` class does not handle the pre-processing of the data directly, to allow for greater flexibility. Therefore, the necessary pre-processing is handled by other scripts elsewhere. Since the pipeline works directly with the input and output data, it may be extended to identify models of systems beyond the quadrotor.

IV. Data acquisition and processing

All outdoor flight data[††] is obtained through the manual piloted flight of the MetalBeetle, composed of both line-of-sight (LOS) and first-person-view (FPV) flights. To reiterate, piloted flight allows for more flexibility in the performed manoeuvres and ensures better safety while pushing the quadrotor to its limits, especially when flying outdoors.

A summary of the procedures for experimental data collection for the outdoor flights is given in section IV.A. The raw flight data is subsequently processed to remove sensor biases, filter noise, and prepare the data for the system identification pipeline. For each of the forces and moments, relevant excitations are extracted from the processed data. This subset is then further split into validation and training data sets to facilitate model identification. This raw data processing procedure is outlined in section IV.B

A. Experimental data collection

Outdoor flights for the MetalBeetle were conducted in a large open-field shielded by trees approximately 200 *m* long and 200 *m* wide with a few trees at the center. For these flights, the altitude was only physically constrained by the range limits of the on-board receivers. These large outdoor spaces allowed for more aggressive flight (e.g. barrel

[††]Indoor flights were also conducted but, to limit the scope of the paper, data acquisition and processing procedures of the indoor flights are given in section VIII.D.

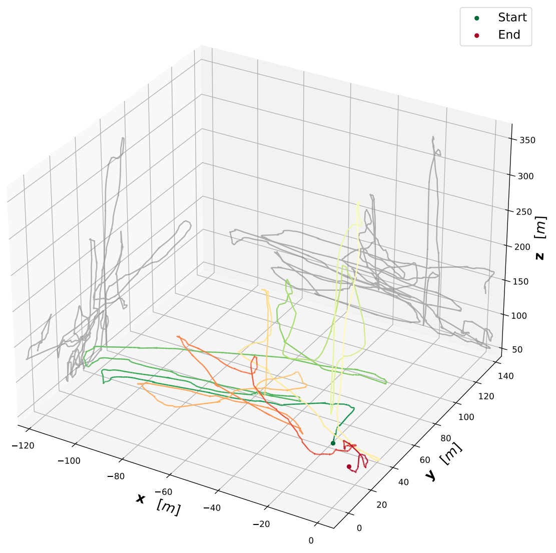


Fig. 6 Illustrative example of the three-dimensional trajectory flown by the MetalBeetle during one of the outdoor line-of-sight (LOS) flights. Also shown, in grey, are the two-dimensional traces of this flight in the $x - z$ and $y - z$ planes for reference.

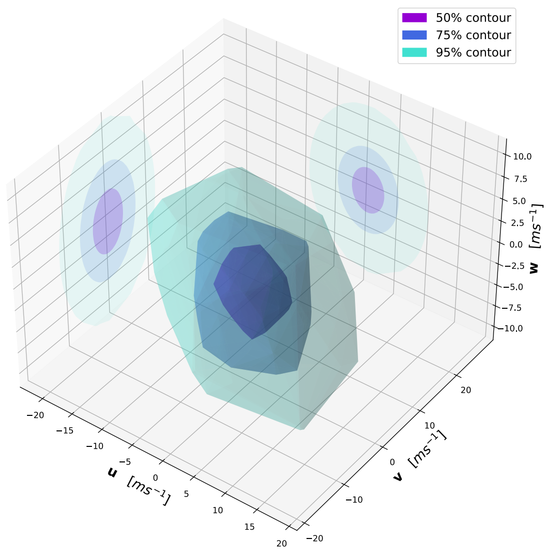


Fig. 7 Velocity phase portrait of the *outdoor* MetalBeetle flights with contours encompassing velocities 50%, 75%, and 95% away from the mean.

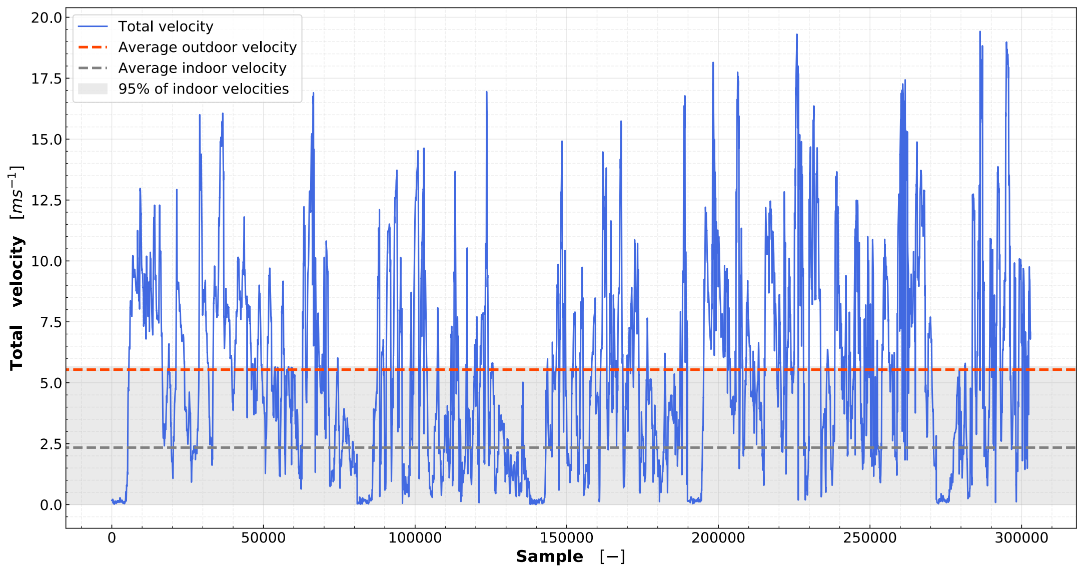


Fig. 8 Total velocity profile across all outdoor flights of the MetalBeetle, as measured by the GPS module. Also shown by the orange dotted-line is the average velocity. For reference, the average velocity achieved through the indoor flights of the MetalBeetle along with where 95% percentile of the highest velocities attained indoors are also shown by the grey dotted-line and grey region respectively.

rolls and punch-outs) and consistent high speed flight of around 10 ms^{-1} as shown by the velocity phase portrait of the MetalBeetle, fig. 7. In fig. 7 the velocities are shown as contours encompassing 50%, 75% and 95% of the closest measurement data to the mean to provide insight on where these velocities are concentrated.

All state information, including the rotor speeds, is obtained from on-board sensors and is logged at 500 Hz. The position and velocity estimations are taken from the on-board GPS module with a fluctuating refresh rate of between $1 - 10$ Hz. This GPS-derived information is then up-sampled and aligned with the IMU data through the timestamps of the GPS updates in the logging procedure of BetaFlight.

Furthermore, as the GPS module acts as the primary source of velocity information, the presence of wind can severely affect the true velocity measurements. While this may be mitigated by flying both with and against the wind in a single flight (such that the stationary wind effects average out) it is still detrimental to the development of high fidelity models. Moreover, such flight cannot eliminate the effects of gusts or instantaneous changes in wind direction. Without a reliable way to estimate the relative velocity of the quadrotor, and due to poor weather conditions on the flight days, flights for the MetalBeetle were contaminated with wind (average wind speed of around 6 ms^{-1} and gusts of up to 10 ms^{-1}). This strongly urges the development of more reliable methods to determine the air speed of the quadrotor.

Both line-of-sight (LOS) and first-person-view (FPV) flights were conducted outdoors. Prior to take-off, the quadrotor was switched on and held stationary for a few seconds to facilitate the estimation of the sensor biases for raw data processing. Note that BetaFlight supports a so-called 'air' mode wherein the rotors spin at an idle rate with the throttle set to zero. Therefore, the rotors are spinning at this idle rate as soon as the quadrotor is armed and no input to the throttle is given. This mode was employed since it ensures that the rotors remain spinning even when the throttle is cut during flight, such that control of the quadrotor can be maintained during aggressive flight.

The LOS flights are aimed at exciting the forces and moments along each axis through simple, repeatable, manoeuvres. As such, LOS manoeuvres consisted of sinusoidal step inputs (e.g. backward-and-forward, side-to-side, and ascent-and-descent manoeuvres) and rapid pulses in throttle, roll, pitch and yaw. The sinusoidal manoeuvres were conducted by repeatedly moving back-and-forth between the extremes of the outdoor space along a given axis while keeping deviations in all other axes to a minimum. For instance, manoeuvres along the x-axis involved pitching forward as far as possible and increasing thrust to move forward while maintaining altitude until one end of the field is reached then immediately pitching backwards to return to the opposite end of the field. This sequence was repeated at least few times, but varied in number across flights. Likewise, manoeuvres along the y-axis were conducted in a similar manner. Manoeuvres along the z-axis involved setting the throttle to maximum to ascend quickly, and reducing it to 'idle' to descend rapidly. Again this sequence was repeated a few times to complete the manoeuvre. Additionally, throttle 'pulses' where also conducted, which involved the rapid 'flicking' of the throttle in a manner analogous to ON-OFF control. Note that not all these manoeuvres were included in a single flight plan, but were rather spread over numerous flights. For example, an arbitrary flight may involve x- and y- axis manoeuvres and the subsequent flight includes y- and z-axis manoeuvres. Therefore, in aggregate, multiple flights containing these sinusodial step inputs along each of the axes were conducted for the MetalBeetle such that similar manoeuvres may be found across many of the flights. An example trajectory of one of the outdoor LOS flights of the MetalBeetle is given in fig. 6. Indeed, fig. 6 only emphasizes the space needed to adequately achieve high-speed flight.

The FPV flights are geared towards aggressive manoeuvring and involved punch-outs, (front and back) flips, and barrel rolls. Slalom-like flight between the trees located in the middle of the field was also conducted. As is the case with such 'freestyle' flying, manoeuvres are less repeatable. However, the ultimate goal is for the development of models valid for such flight, irrespective of the exact manoeuvre conducted. Ideally, they are able to adequately capture the underlying processes of these manoeuvres. Moreover, through these flights, the MetalBeetle is recorded to achieve a top speed of 19 ms^{-1} and consistently hits 10 ms^{-1} as shown in fig. 8. Note that, due to the wind, these GPS measurements may not be entirely accurate of the true speeds achieved. However, they give an indication of the attainable, and commonly flown, speeds outdoors. The average velocity flown outdoors is already double that of the indoor flights of the MetalBeetle and is only slightly below the 95% percentile of the highest velocities achieved indoors. Readers interested in the indoor analysis are directed to section VIII.D.

B. Raw data processing

Following data acquisition, the raw data is processed in preparation for use with the system identification pipeline. The processing of the raw flight data is conducted in Python 3.7 with self-written scripts. Note that measurements coming from the on-board sensors (i.e. originating from IMU, ESC, and GPS) are filtered by the flight controller before logging. These measurements are therefore pre-filtered prior to any processing described in this section. A dynamic (i.e. changes with respect to the throttle level) low-pass filter is employed by BetaFlight to remove high-frequency noise in the accelerometer data with cut-off frequencies ranging from $50 - 170\ Hz$. A more advanced filtering scheme (i.e. adaptive and involving numerous filters such as low-pass and notch filters) is employed to filter the gyroscopic data since it is essential for maintaining control of the quadrotor through the PID controllers. As such, the logged data represents the measurements after all this filtering is applied and does not reflect the true raw data produced by the sensors. Examples of these pre-filtered logged data measurements for the gyroscope are given by fig. 9 and fig. 10 for hovering and arbitrary

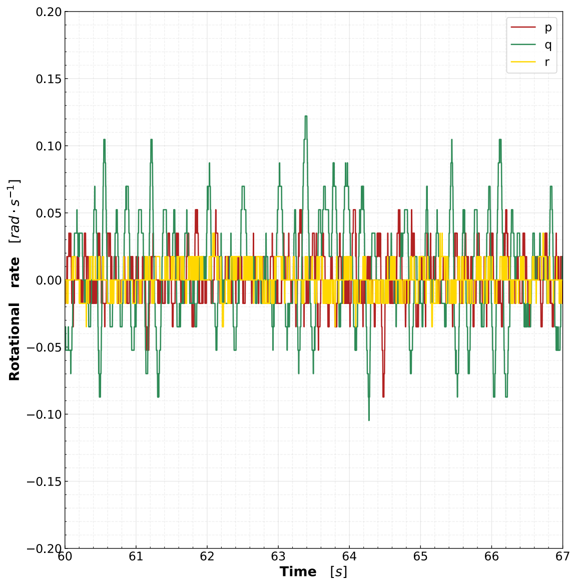


Fig. 9 Sample of the raw logged rotational rate measurements during hovering flight of the MetalBeetle.

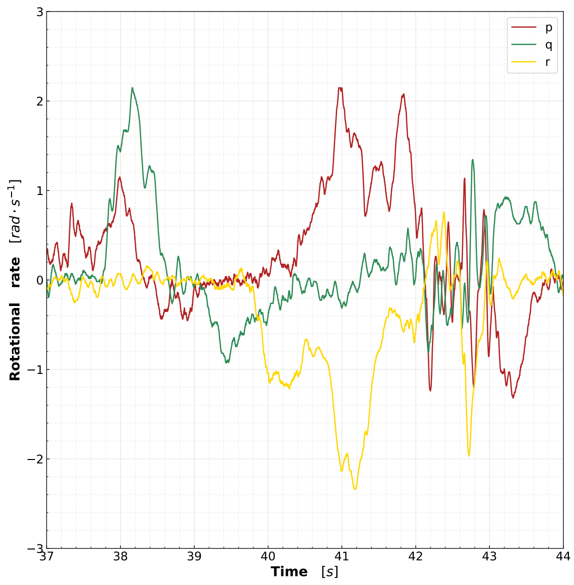


Fig. 10 Sample of the raw logged rotational rate measurements during arbitrary flight of the MetalBeetle.

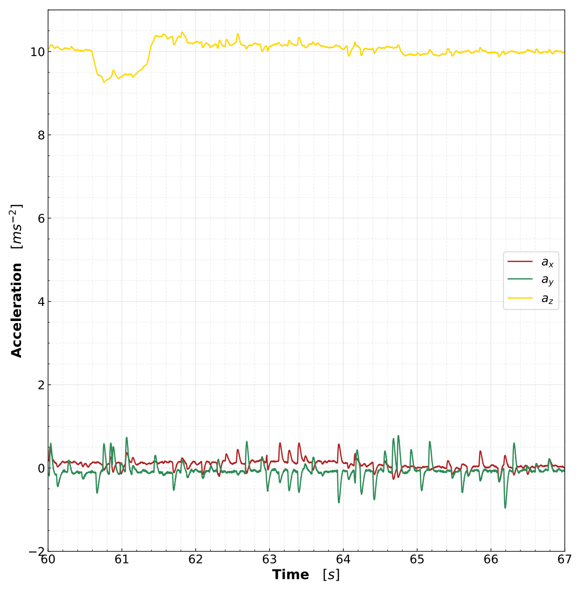


Fig. 11 Sample of the raw logged acceleration measurements during hovering flight of the MetalBeetle.

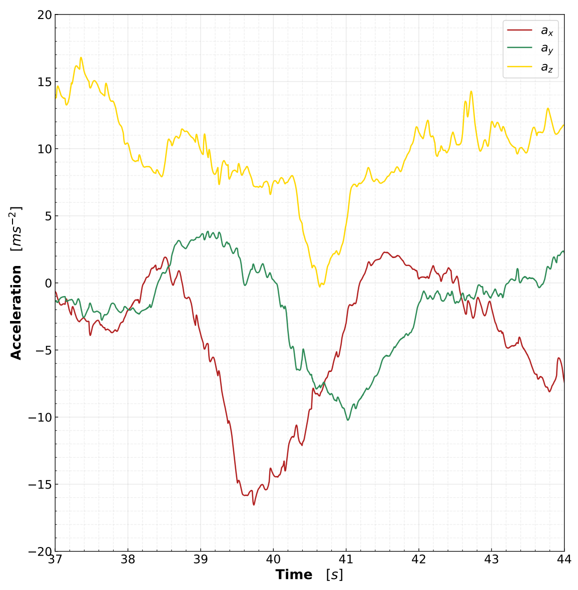


Fig. 12 Sample of the raw logged acceleration measurements during arbitrary flight of the MetalBeetle.

flight respectively. Likewise, fig. 11 and fig. 12 depict the accelerometer measurements for hovering and arbitrary flight respectively. Indeed, the pre-filtering of BetaFlight is apparent through rather noise-free signals.

Moreover, while the on-board data is recorded at 500 Hz, this sampling rate was found to be inconsistent wherein the true logging rate varied from 470 – 510 Hz. Nonetheless, the logging module of BetaFlight handles the synchronization of all the sensor information, including GPS. Hence, the dropped, or added, frames are consistent across all the measurements. Consequently, regardless of indoor or outdoor flight, the processing pipeline first resamples the on-board data to 500 Hz using a cubic spline interpolation to ensure consistency.

1. Outdoor specific data pre-processing

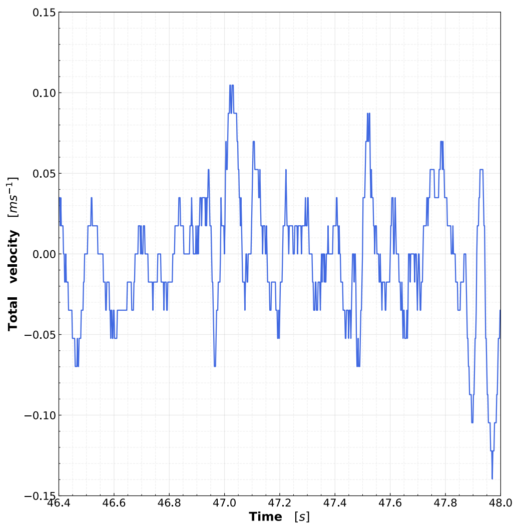
Even though the all the state information is measured, and synchronized, by the on-board sensors for the outdoor flights, there are still some outdoor-specific processing procedures employed to manage the GPS-based data. A sample of the raw logged GPS velocity measurement data is depicted in fig. 13 for a period of stationary flight. Moreover, the PSD of the GPS velocity measurements is shown in fig. 14.

As aforementioned, the GPS measurements are inconsistent and updated slowly in comparison to the rest of the on-board data (i.e. $1 - 10$ Hz for GPS versus ≈ 500 Hz for IMU and ESC). This culminates in a box-like signal for the GPS velocity measurements as shown in fig. 15. Therefore, as the on-board data is re-sampled using a cubic spline interpolation for sampling consistency, ringing artefacts are introduced into the velocity (and position) measurements through Runge's phenomenon as shown in fig. 15.

Therefore, to mitigate this, the velocity measurements are further filtered using a Savitzky-Golay filter. The parameters which resulted in the most visually accurate results are a window size of 151 samples with a polynomial order of 3. While the results are still imperfect, they produce a more faithful (continuous) representation of the underlying velocity signal than the re-sampling alone as illustrated by fig. 15. Indeed, it is possible to first apply this Savitzky-Golay filter to the GPS velocity prior to the re-sampling. However, the resultant signal is similar to that obtained when filtering is applied after the resampling.

2. General data processing

The flight data is then passed through an Extended Kalman Filter (EKF) to estimate the accelerometer and gyroscope biases. The EKF is constructed following the procedure of Armanini et al. [50], who define quadrotor-specific EKF equations. However, these process equations do not consider much of the aerodynamic effects targeted by the present



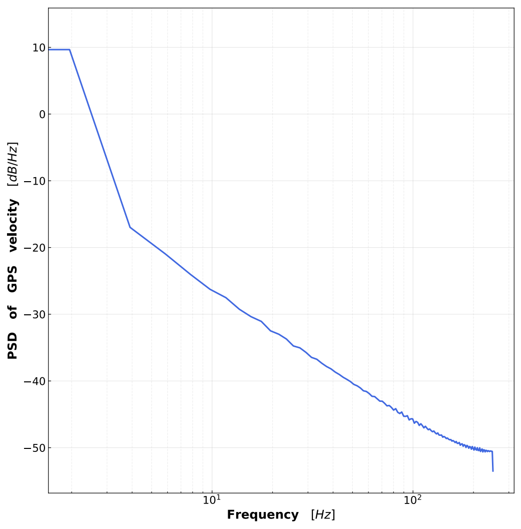


Fig. 13 Sample of the raw logged GPS velocity measurements during stationary flight of the MetalBeetle.

Fig. 14 Power spectral density of the GPS velocity measurements of the MetalBeetle.

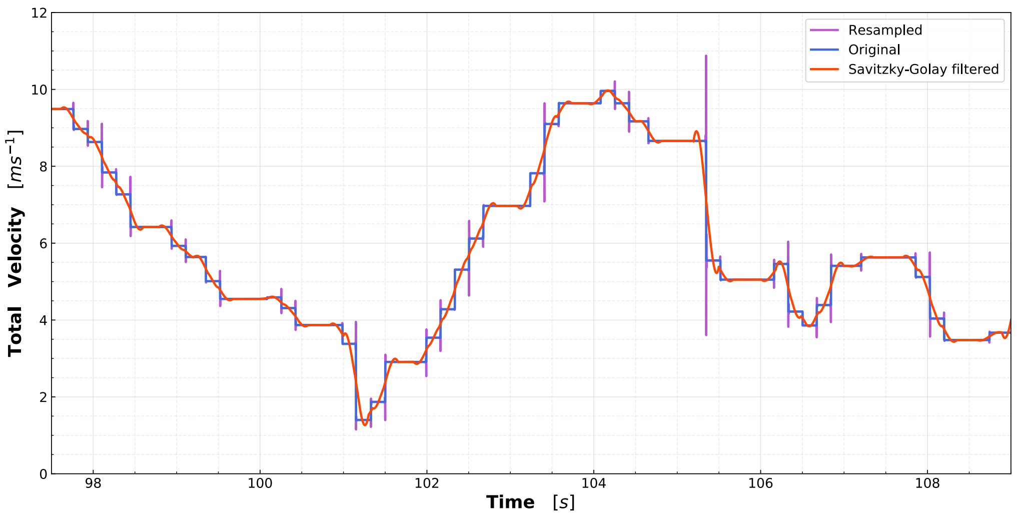


Fig. 15 Illustrative example of the box-like GPS velocity measurements, subsequent artefacts introduced by re-sampling, the filtering of this signal to remove said artefacts for an arbitrary outdoor flight of the MetalBeetle

work. In fact, the estimated biases were observed to vary considerably during flight, likely representing the un-modelled dynamics in the process equations. Hence, the sensor biases are only reliably estimated during the stationary period after arming but immediately preceding take-off.

In accordance with the processing procedure of Sun et al. [8], the unbiased acceleration and rotational rates are then further filtered by a forth-order low-pass Butterworth filter to remove noise. However, in the current research, different cut-off frequencies are elected to capture more of the high-frequency dynamics. In fact, much of the rotor dynamics occur in the 30-80 Hz region. Therefore, the 5 Hz and 16 Hz cut-off frequencies employed by Sun et al. [8] for the accelerometer and gyroscope measurements respectively are much too low. Here, a cut-off frequency of 100 Hz is applied to both the accelerometer and gyroscope instead. The corresponding PSDs are illustrated in fig. 16 and fig. 17 for the accelerometer and gyroscope respectively. Note that 100 Hz is used here since much of the BetaFlight filtering already eliminates frequencies beyond 100 Hz, thus any information contained in the frequencies beyond 100 Hz is

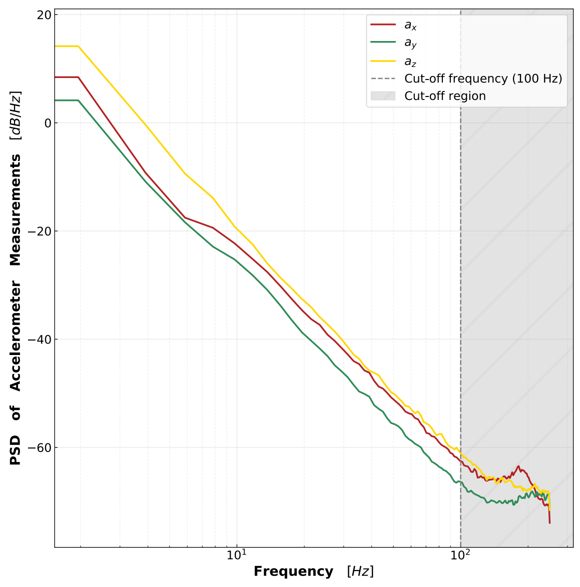


Fig. 16 Power spectral density of the accelerometer (MPU6000) measurements of the MetalBeetle. The grey-dotted line denotes the chosen cutoff frequency.

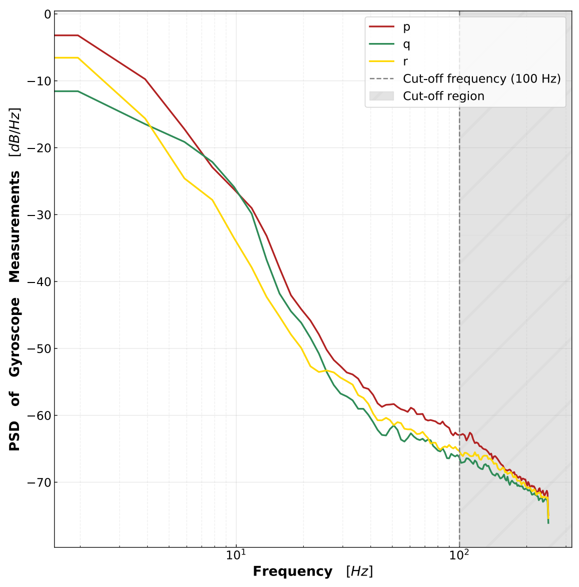


Fig. 17 Power spectral density of the gyroscope (MPU6000) measurements of the MetalBeetle. The grey-dotted line denotes the chosen cutoff frequency.

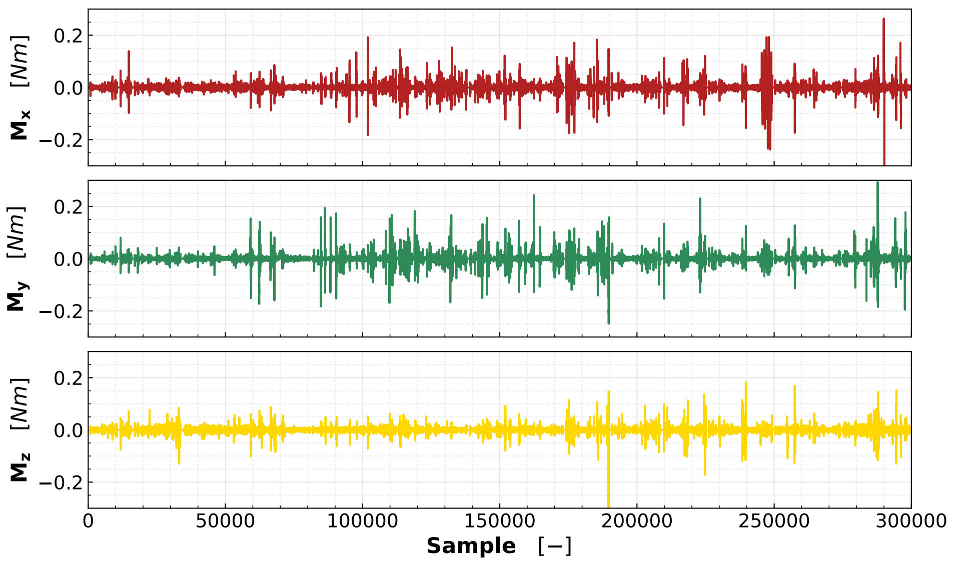
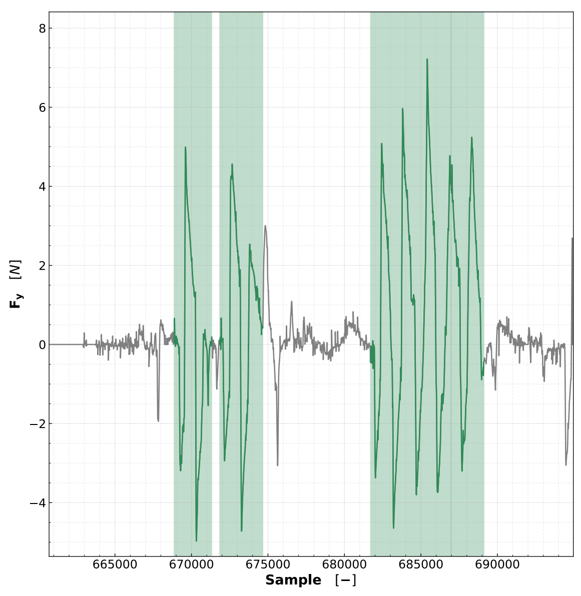


Fig. 18 Moment measurements of the outdoor MetalBeetle flights.

mostly lost. The filtered and unbiased acceleration measurements are subsequently used to compute the body forces (through eq. (3)), while the filtered and unbiased rotational rates facilitate the moment computations (through eq. (2)).

Subsequently, the induced velocity, v_{in}, is calculated as it is not measured directly by the quadrotor. Indeed, eq. (16) may be used to estimate the induced velocity, however, given that this is a forth order polynomial, multiple solutions (i.e. roots) exist for v_{in}. Here, it is assumed that the induced velocity is similar, in magnitude, to the total velocity. Hence, the root which lies closest - in absolute terms - to the current velocity is taken as the true induced velocity.

Prior to model identification, the input states are normalized following the methods outlined in section III.B. Through this normalization, the normalized total rotor speed, normalized advance ratios, and normalized control moments are



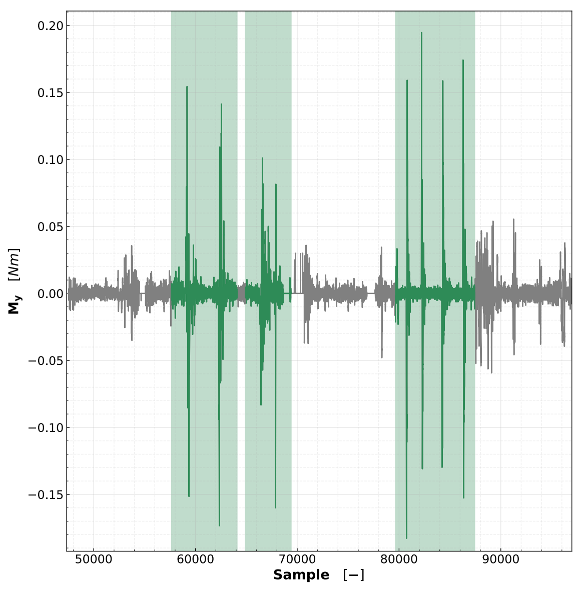


Fig. 19 Snippet of the excitation isolation algorithm, as applied to arbitrary F_y data of the MetalBeetle, highlighting the regions of identified excitation.

Fig. 20 Snippet of the excitation isolation algorithm, as applied to arbitrary M_y data of the MetalBeetle, highlighting the regions of identified excitation.

obtained. Likewise, the quadrotor's attitude is effectively normalized by taking their trigonometric identities. This then completes the defined input state vector, eq. (25).

Once all the data has been processed, a partition is made based on the the flights of the quadrotor. Certain flights are elected for training the models while others are reserved purely for testing. The split is made such that excitations along each of the axes are present in both the training and testing (i.e. validation) sets. It should be noted that the quadrotor models are never exposed to the testing flights, and are only evaluated against them once training is complete. Due to the aforementioned variability in manoeuvres between flights, entire flights are preferred for partitioning the training and testing data sets in order to keep manoeuvres continuous. In any case, it is held that identified models should be able to adequately capture the dynamics behind any of the flights. For all flights, data is truncated to the region after take-off and before landing. Since some flights ended in crashes, the last five percent of the data is discarded.

Furthermore, during identification, the training data set is further constrained to the regions of excitation for the relevant force or moment. For instance, for models of F_x, only regions of the training data set with excitations along the x-axis are used for model identification. Indeed, there is a difference in responses characteristics between the force and moment models. Therefore, for the forces, regions of excitation were determined by examining where the local variance in force response is larger than the global variance scaled by some threshold (set through trial-and-error to 0.15). Whereas, given the sparsity of the moment excitations (refer to fig. 18), the prominent peaks in the signal are interpreted as excitations. These peaks are identified using the `scipy.signal.find_peaks` function applied to the absolute value of the moment measurements. Since useful dynamics may still be obtained by observing the states immediately preceding, and following the completion of, the isolated excitations, data samples before and after the excitations are also included in the identification subset. An example of the results of this excitation isolation algorithm are illustrated in fig. 19 for the F_y and fig. 20 for the M_y measurements of the MetalBeetle.

V. Model Identification Results

After all the flight data has been processed, outdoor models of the chosen quadrotor platforms may be identified[‡‡]. The performance of the identified models is assessed through their accuracy and quality of prediction intervals (PIs). The accuracy is measured through the normalized[§§] root-mean squared error (NRMSE) and the coefficient of determination

[‡‡]Note that equivalent indoor models of the MetalBeetle may be found in section VIII.D

[§§]The RMSE is normalized with respect to the range of the target data to facilitate a comparison between the different identification techniques.

(R^2) with respect to both the entire data set and the testing data subset. The quality of the identified models' PIs is evaluated through the coverage probability ($PICP$) and normalized mean PI width ($MPIW$). The identified model PIs are constructed using a 95% confidence level and follow from the procedures in **Part I: On prediction intervals**. As such, valid models should maintain a $PICP \geq 95$. Moreover, some illustrative examples of the models' predictions on an entire test subset flight are shown here. However, for brevity, figures and tables (e.g. of polynomial regressors) summarizing most of the identified models are shown in the appendix (namely, sections VIII.A.3 to VIII.A.5).

It is expected that the identified outdoor MetalBeetle models' performance will be adversely affected by the wind contamination despite the aforementioned efforts made to mitigate its effect (e.g. flying both again and with the average wind). Nevertheless, the outdoor model results may still contain useful information and could provide insight on the necessity and configurations for airspeed sensors in future experiments. The performance metrics of the identified outdoor force models of the MetalBeetle can be found in table 2.

Due to the presence of non-stationary effects in outdoor flights, it is perhaps unsurprising the identified ANN force models of the outdoor MetalBeetle mostly accommodate best accuracy between the identification techniques. However, the associated PIs fall short of the $PICP \geq 95$ condition, rendering the models unreliable and potentially invalid. The identified ANN F_x model of the outdoor MetalBeetle yields the poorest performance of the ANN models ($R^2 = 0.2053$, $PICP = 90.8 < 95$, and $MPIW = 20.8\%$) while the identified ANN F_y model produces slightly better results ($R^2 = 0.2913$, $PICP = 92.3 < 95$, and $MPIW = 18.1\%$). Indeed, both the F_x and F_y ANN models show poor accuracy. Peculiarly, the identified ANN model for F_z hosts better performance, bordering acceptable metrics ($R^2 = 0.7700$, $PICP = 93.3 < 95$, $MPIW = 15.6\%$). The improved performance may be due to the fact that the (average) wind is mostly parallel to the ground, and therefore affects the F_x and F_y models more than the F_z models. This is especially true for the sinusoidal ascend-descend manoeuvres. These performances only accentuates the adverse influence of unknown wind.

Similarly, the performances of the identified polynomial outdoor force models parallel those of the equivalent ANN models. The selected model regressors for the normalized F_x, F_y, and F_z models may be found in tables 5 to 7 in

Table 2 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid force models for the MetalBeetle (Outdoor flight). The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Fx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0645	0.0747	0.3971	0.2053	92.8196	90.7920	19.8318	20.8188
Poly	0.0765	0.0846	0.1519	-0.0197	95.4948	94.9728	46.9551	46.0148
Hybrid	0.0714	0.0800	0.2608	0.0868	97.3339	96.6688	51.3899	50.8426
Fy								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0567	0.0627	0.4616	0.2913	93.6676	92.2540	17.6020	18.1279
Poly	0.0622	0.0619	0.3519	0.3085	96.7452	96.3193	36.7955	36.1483
Hybrid	0.0587	0.0620	0.4221	0.3065	98.0834	97.4967	41.3574	40.8746
Fz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0414	0.0485	0.8262	0.7700	95.3711	93.3252	14.5807	15.6362
Poly	0.0559	0.0593	0.6829	0.6563	95.0270	94.4171	34.5638	33.9786
Hybrid	0.0467	0.0508	0.7788	0.7471	97.7102	96.9595	37.3214	36.8635

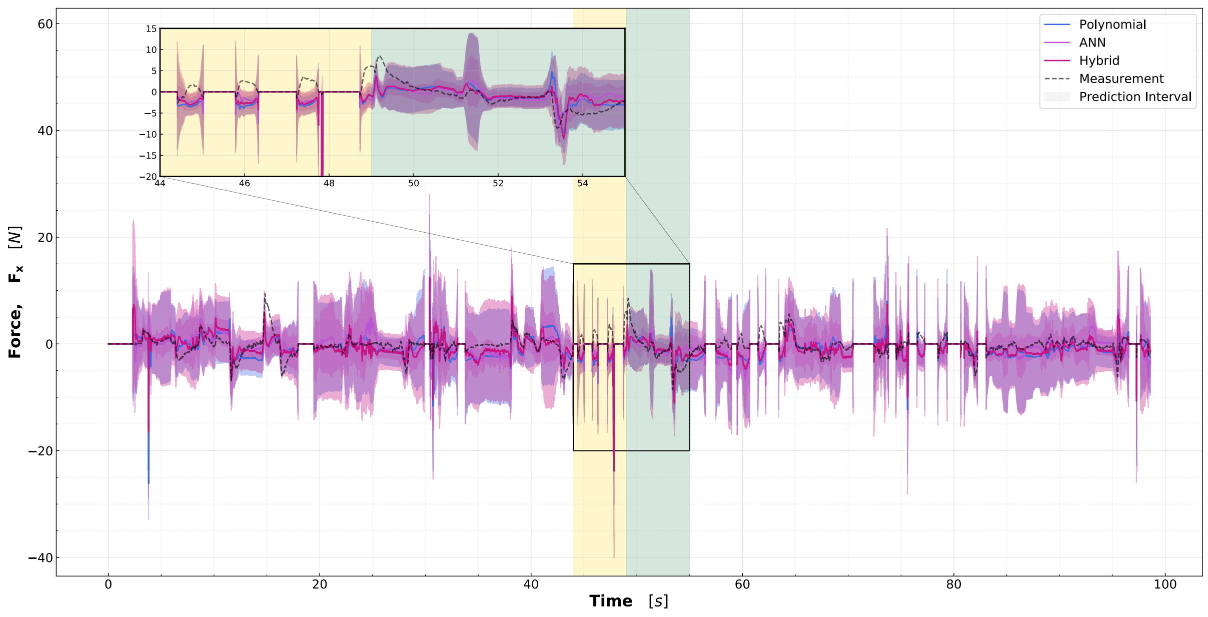


Fig. 21 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_x for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured F_x response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\ ms^{-1}$ with gusts of up to $\approx 10\ ms^{-1}$). The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind.

section VIII.A.5 respectively. Also in the appendix are plots of the polynomial error residuals (section VIII.A.1) and autocorrelation thereof (section VIII.A.2). From fig. 67, the polynomial error residuals are zero-mean and appear to be generally contained within the $1-\sigma$ bounds. The corresponding autocorrelation these residuals, given in fig. 69, imply white noise (i.e. no correlation). Therefore, it may be said that the OLS assumptions are satisfied. The identified F_x polynomial model of the outdoor MetalBeetle presents an unacceptable accuracy ($R^2 = -0.0197$) with impractically wide, but valid, PIs ($PICP = 95.0$, $MPIW = 46\%$). In fact, these large PIs are a common feature of the outdoor models, reflecting the large uncertainty involved in outdoor flight. Note, however, that the PIs found here are likely inflated by the unknown wind experienced during flight. The identified polynomial model of F_y hosts better performance than the F_x model through a higher accuracy ($R^2 = 0.3085$) and narrower, yet valid, PIs ($PICP > 95$, $MPIW = 36.2\%$). As with the ANN models, the identified F_z polynomial model of the outdoor MetalBeetle displays the highest accuracy of the polynomial force models ($R^2 = 0.6563$) although with marginally unreliable PIs ($PICP = 94.4 < 95$, $MPIW = 34.0\%$).

The identified hybrid force models of the outdoor MetalBeetle command improvements over their underlying polynomial models, particularly in terms of extending PI validity. However, gains in accuracy are not as compelling as hypothesized during the conceptualization of the hybrid approach. The hybrid model of F_x manages to recover some modelling accuracy over the polynomial model but still fails to fit most of the measured F_x ($R^2 = 0.0868$). Through wider PIs, the hybrid model comfortably meets the PICP criterion ($PICP > 95$, $MPIW = 50.8\%$). Conversely, the hybrid F_y model of the outdoor MetalBeetle fails to improve the accuracy of the underlying polynomial ($R^2 = 0.3065$) while extending the, already much too wide, PIs ($PICP > 95$, $MPIW = 40.9\%$). The most pronounced increase in accuracy over the polynomial model is obtained by the hybrid F_z model ($R^2 = 0.7471$) with now valid PIs ($PICP > 95$, $MPIW = 36.9\%$). Taken together, the improvements afforded by the hybrid models are not as pronounced as expected. While they do appear to promote valid PIs, this is somewhat irrelevant as these PIs are already impractically wide and the associated model predictions unacceptably poor. Subsequently, these results provide little support for the use of hybrid models, especially over their constituent system identification techniques.

An illustrative example of the predictive performances of these identified force models is depicted in fig. 21. Here, the predictions with respect to F_x are shown for an unseen validation LOS flight. Illustrations of the F_y and F_z model predictions for the same flight may be found in section VIII.A.4. The F_x response of the outdoor MetalBeetle in fig. 21

is peculiar in that there are sudden periods of zero force scattered throughout the response. This is occurs when the rotor speeds fall below the idle $eRPM$ (= 750) enforced in the processing scripts to avoid singularities and high magnitude terms during normalization. Despite this cut-off, there are still some artifacts present in the data produced by the normalization with low $eRPM$ values. Take, for instance, the spikes immediately following, and immediately preceding, the zero-force regions in the yellow highlighted region of fig. 21. The prevalence of these low $eRPM$ values is likely due to the aggressive manoeuvres conducted, for which the throttle is cut often. Moreover, the effects of the strong wind present throughout the flight may also contribute to this through fluctuating rotor speeds induced by the flight controller.

In fact, there is a clear difference in the characteristics of the measured force when flying into, or away from, the wind. In fig. 21, the yellow highlighted region denotes backward flight (i.e. along negative x) into the incident wind while the green region represents forward flight (i.e. along positive x) with the wind. When flying into the incident wind, the component of thrust along the horizontal (i.e. parallel to the wind) needs to be increased to counteract this wind. Assuming a fixed pitch angle (which is the case when the pitch is at its maximum, like in this manoeuvre), only the thrust vector itself may be increased to mitigate the wind. This will induce a climb for the quadrotor, since the vertical component of motion is less affected by the incident wind (which flows mostly parallel to the ground). The pulse-like nature of this response stems from the fact that the throttle was periodically pulsed to maintain a somewhat constant altitude while also exciting the dynamics. Paradoxically, this ascent stimulates a (brief) positive acceleration along the *body* x-axis, hence the positive measured F_x in the yellow region of fig. 21 during the pulses. Indeed, these positive pulses correspond exactly to increases in total rotor speed. The divergence of the measured force and predicted force likely emanates from the mismatch in velocities and rotor speeds experienced during this portion of the flight. Indeed, the measured velocities from the GPS only track the ground speed, and not the airspeed, of the quadrotor. This velocity discrepancy also adversely affects the models' judgement of the relation between thrust level and (absolute) velocity. For example, when flying into the wind the quadrotor experiences a lower than expected velocity for a given thrust level than when flying without wind.

Conversely, when flying with the wind, as in the highlighted green region of fig. 21, the quadrotor flies with a higher velocity than expected for a given thrust level due to the pushing of the wind. Despite this, the models are able to capture the measured force much more effectively when flying with the wind than against it. This implies that there are some rotor inflow effects, depending on the direction of this inflow to the propeller, which are important for the force modelling. The sudden widening of PIs that occurs between 51 and 52 seconds are of note since it implies an abrupt uncertainty in the predictions. In fact, this coincides with a rapid decrease in throttle to reduce altitude and may thus reflect the uncertainties related to the inflow of wind into the underside of the rotors (i.e. potentially some auto-rotation effects). The rapid change in force starting at 53 seconds corresponds to the pitching back of the quadrotor to brake and change direction. Interestingly, the identified models successfully reproduce the forces involved with this transition from forward to backward flight, but fail to reproduce the measured forces associated with the opposite transition (see measured forces at around 49 seconds in fig. 21). This again implies that there are significant wind interaction effects and potentially rotor wake effects. Overall, the identified F_x models are able to loosely follow the measured forces, with the exceptions of regions where there is a perceived mismatch in input states.

Nevertheless, despite the contamination of wind, the outdoor models are able to successfully capture the forces involved with aggressive manoeuvring. Figure 22 illustrates the models' predictions of F_z during various aggressive manoeuvres for an FPV flight. Note that F_z is shown here since most of the manoeuvres influence F_z. A punch-out manoeuvre - whereby the throttle is increased to the max - is conducted in the highlighted yellow region at the start of the flight. It is clear that, while not perfect, the models' predictions are in line with the measured force. In any case, the force is mostly contained within the prediction intervals. The polynomial and hybrid models give the most faithful reproductions of this manoeuvre. In contrast, the ANN models represent the barrel-roll and (front and back) flip manoeuvres the best. The barrel rolls are highlighted in green, front flips in red, and back flips in orange in fig. 22. It is clear that the identified models are able to accurately predict the forces induced by these manoeuvres well, especially the barrel rolls. Although the polynomial clearly performs the worst for these manoeuvres, it is still able to capture the majority of the dynamics. Interestingly, there are some oscillations predicted for the back flips, which do not occur for the front flips. The exact reasons for this are unknown but may relate to the interaction of the rotors with the quadrotor's disturbed air during the back flip manoeuvre. The oscillations are then induced by the fluctuating rotor speeds maintaining control and appear in the models due to their reliance on the total rotor speed. Moreover, the quadrotor achieved a recorded (ground) flight speed of 19 ms^{-1} for the period between 29 and 30 seconds. Indeed, the identified models are able to capture the forces involved during this high speed flight. The ability of the identified models to capture these aggressive manoeuvres and high-speed flight, despite the persistent unknown wind, is a significant outcome and demonstrates the potential of the system identification pipeline and techniques used therein.

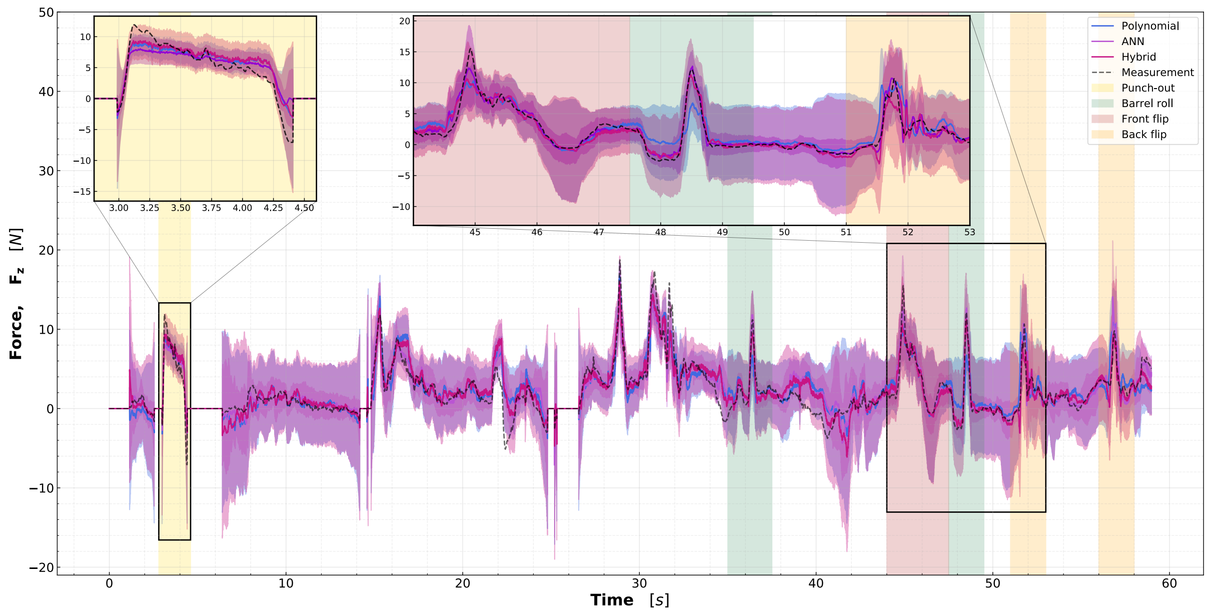


Fig. 22 Demonstration of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_z for flight 37 of the MetalBeetle involving several aggressive manoeuvres. Also shown is the measured F_z response (black dotted line). This flight was a first-person-view outdoor flight in windy conditions (wind speed $\approx 6\ ms^{-1}$ with gusts of up to $\approx 10\ ms^{-1}$). Highlighted in various colors are the conducted aggressive manoeuvres including punch-outs, barrel-rolls, front-flips, and back-flips.

The performance metrics of the identified outdoor moment models of the MetalBeetle are summarized in table 3. Note that, due to the sparsity of moment responses (refer to fig. 18), the moment performance metrics are taken with respect to the regions localized around areas of moment excitation to avoid evaluating model accuracy predominately with respect to noise. The moment model performances with respect to the full data may be found in table 4 in section VIII.A.3. Interestingly, the PIs are narrower for the moment models than for the force models. Indeed, the effects of wind likely influence the moments less than the forces since the moments are primarily excited during changes in direction. Thus, the comparatively large PIs found for the force models are probably due to the effects and uncertainties induced by the unknown wind contamination.

The identified ANN moment models of the MetalBeetle present mediocre modelling accuracy with unreliably narrow PIs. This performance is exemplified through the identified rolling moment, M_x, model ($R^2 = 0.3158$, $PICP = 87.5 < 95$, and $MPIW = 4.7\%$). The pitching moment, M_y, model of the outdoor MetalBeetle boasts slightly better performance ($R^2 = 0.3744$, $PICP = 90.2 < 95$, $MPIW = 5.9\%$). Although an improvement, the outdoor ANN M_y model still fails to meet the $PICP \geq 95$ criterion with invalid PIs. In comparison to ANN M_x and M_y models, the identified ANN yawing moment, M_z, model hosts the worst performance ($R^2 = 0.1885$, $PICP = 88.6 < 95$, and $MPIW = 6.9\%$). This may be due to the fewer, and less pronounced, M_z excitations in the data. Compare, for example, the magnitudes and frequency of the M_z response to the M_x and M_y responses in fig. 18.

The selected regressors of the identified polynomial moment models, along with their associated coefficients, covariances, and cumulative contribution to the models' accuracy (in terms of R^2) are summarized by tables 8 to 10 in section VIII.A.5. Also in the appendix are plots of the polynomial moment error residuals (fig. 68) and autocorrelation thereof (fig. 70). Indeed, the OLS assumptions of zero-mean white noise error residuals appear to be satisfied.

While the identified outdoor polynomial rolling moment, M_x, model experiences an improvement in modelling accuracy over its ANN counterpart ($R^2 = 0.4906$), it still suffers from unreliable and wider PIs ($PICP = 88.2 < 95$, $MPIW = 8.6\%$). Likewise, the identified polynomial pitching moment, M_y, sees a boost in accuracy ($R^2 = 0.4518$) with wider, although marginally unreliable, PI metrics ($PICP = 94.0 < 95$, $MPIW = 16.9\%$). In contrast, the identified polynomial yawing moment, M_z, model presents both a poorer modelling accuracy ($R^2 = 0.0702$) and disappointing PI metrics ($PICP = 91.8 < 95$, $MPIW = 15.5\%$) in comparison to its ANN equivalent. Interestingly, the polynomial M_z

Table 3 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid moment models for the MetalBeetle (Outdoor flight) *localized around regions of excitation*. The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Mx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0293	0.0330	0.3920	0.3158	90.6702	87.5393	4.0409	4.6688
Poly	0.0231	0.0285	0.6226	0.4906	92.5027	88.1838	8.7540	8.5669
Hybrid	0.0253	0.0308	0.5454	0.4052	96.9265	95.0168	11.4349	11.3636

My								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0399	0.0431	0.4091	0.3744	91.8470	90.2332	5.8614	5.9279
Poly	0.0386	0.0403	0.4467	0.4518	94.8601	94.0170	18.5884	16.9476
Hybrid	0.0414	0.0438	0.3642	0.3524	97.0208	96.4572	22.2708	20.4219

Mz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0374	0.0401	0.2522	0.1885	89.8208	88.6323	6.0165	6.9141
Poly	0.0372	0.0429	0.2607	0.0702	93.6921	91.7996	14.4381	15.4791
Hybrid	0.0357	0.0409	0.3199	0.1550	95.8735	94.5427	15.9481	17.2706

model is the only outdoor moment model which sees a significant drop in performance between the full and test data sets. This may be symptomatic of over-fitting or, given the better performance of the hybrid model, could represent a lack of approximation power. Indeed, the sparse and low magnitude moment excitation may also contribute to this lackluster performance.

Despite this improvement by the identified hybrid moment models of the outdoor MetalBeetle for M_z, the hybrid moment models generally stimulate a decrease in model accuracy over the underlying polynomial. This is true for both the identified hybrid rolling moment, M_x, and pitching moment, M_y, models ($R^2 = 0.4052$ and $R^2 = 0.3524$ respectively). Such a deterioration in modelling accuracy, especially for outdoor data, comes as a surprise. It was hypothesized that the non-linear approximation capabilities of the ANN would account for some of the modelling deficiencies of the polynomial models, especially with regards to non-stationary effects. Indeed, the effects of wind may be interpreted as strong non-stationary effect. Hence, the failure of the hybrid models to improve on the polynomial performances cast doubt on their utility. Note, however, that the ANN-only moment models generally perform worse than their polynomial equivalents. Hence, the effects of wind on the aerodynamic moments may nonetheless elude to ANN models and, while the results here challenge the merits of the hybrid models, the ANNs themselves may be fitting noise in these measurements instead of useful dynamics. Unlike the underlying polynomials, the hybrid models of M_x and M_y now satisfy the PICP criterion ($PICP = 95$, $MPIW = 11.4\%$ and $PICP > 95$, $MPIW = 20.4\%$ respectively). Conversely, the identified hybrid yawing moment model, M_z, fails to meet this criterion ($PICP = 94.5 < 95$, $MPIW = 17.3\%$) but manages to improve upon the modelling accuracy ($R^2 = 0.1550$) with respect to the underlying polynomial model.

An example of the predictive performances of the identified outdoor pitching moment models of the MetalBeetle with respect to an unseen LOS flight are depicted in fig. 23. Other moment responses for this flight may be found in section VIII.A.4. Oscillating step inputs were given to the pitch angle in the highlighted red region of fig. 23 to excite the pitching moments. For these manoeuvres, the incident wind was perpendicular to the quadrotor's direction of flight. Here, the identified moment models are capable of reproducing the measured force and exhibit a (visually) good fit.

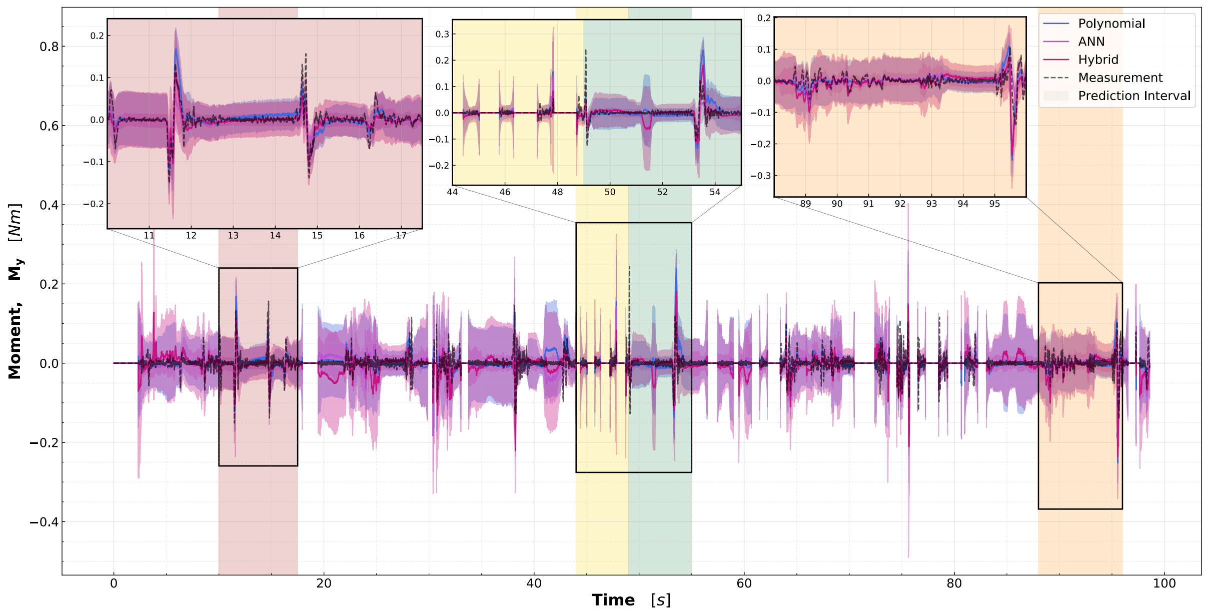


Fig. 23 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_y for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured M_y response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\,ms^{-1}$ with gusts of up to $\approx 10\,ms^{-1}$). Pitching excitations were performed, perpendicular to the incident wind, in the red highlighted region. The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind. Throttle pulses with the quadrotor facing the incident wind were conducted in the orange highlighted region. Due to the wind, some pitching was necessary to avoid drift.

This modelling accuracy deteriorates for pitching manoeuvres conducted parallel to the incident wind [¶¶]. This is evident in the yellow and green highlighted regions, where forward-and-backward flight was conducted for which moment excitations occur when changing direction. The ANN-based models in particular exhibit large deviations from the measured moments which are coincident with changes in total rotor speed. In general, the models' predictions appear to follow the fierce moments more effectively than the more subtle fluctuations succeeding a manoeuvre. However, these models are still able to capture some of the more nuanced moments as shown by the highlighted orange region. Here, throttle pulses along the z-axis were conducted facing the incident wind. To avoid drifting due to the wind the quadrotor's pitch needed to be adjusted during the pulse, hence the measured oscillations. Overall, these results contradict the performance metrics found in table 3 and demonstrate that the identified moment models are able to faithfully reproduce much of the distinct measured moments (in an unseen flight) despite the presence of the wind. However, the more subtle moment oscillations sometimes still elude the identified models. Nonetheless, these results show promise for the identification of outdoor models, especially if measurements of the airspeed are obtained.

As with the force models, aggressive manoeuvres are also captured by the identified outdoor moment models. To illustrate this, consider the rolling moment response of the MetalBeetle during an outdoor flight for which barrel rolls are conducted depicted in fig. 24. Highlighted in green are the conducted barrel roll manoeuvres. Indeed, there are few strong moment responses outside of these manoeuvres, emphasizing the utility of conducting such aggressive flight outdoors to adequately excite the moments. The identified polynomial and hybrid rolling moment models appear to easily capture the dynamics involved with the first barrel roll. However, the ANN model struggles to do so. All model predictions are less faithful for the second barrel roll, wherein they struggle to follow some of the moment oscillations immediately after the completion of the roll. Indeed, even the measured barrel roll responses are different in shape. Through the video logs, it is clear that the second barrel roll is less 'clean' than the first and did not complete fully. This is also obvious through the data where the first barrel roll response is well-defined. It may also be the case that the wind

[¶¶]Recall that the wind speeds during the flight test were approximately 6 ms^{-1} with gusts of up to 10 ms^{-1}.

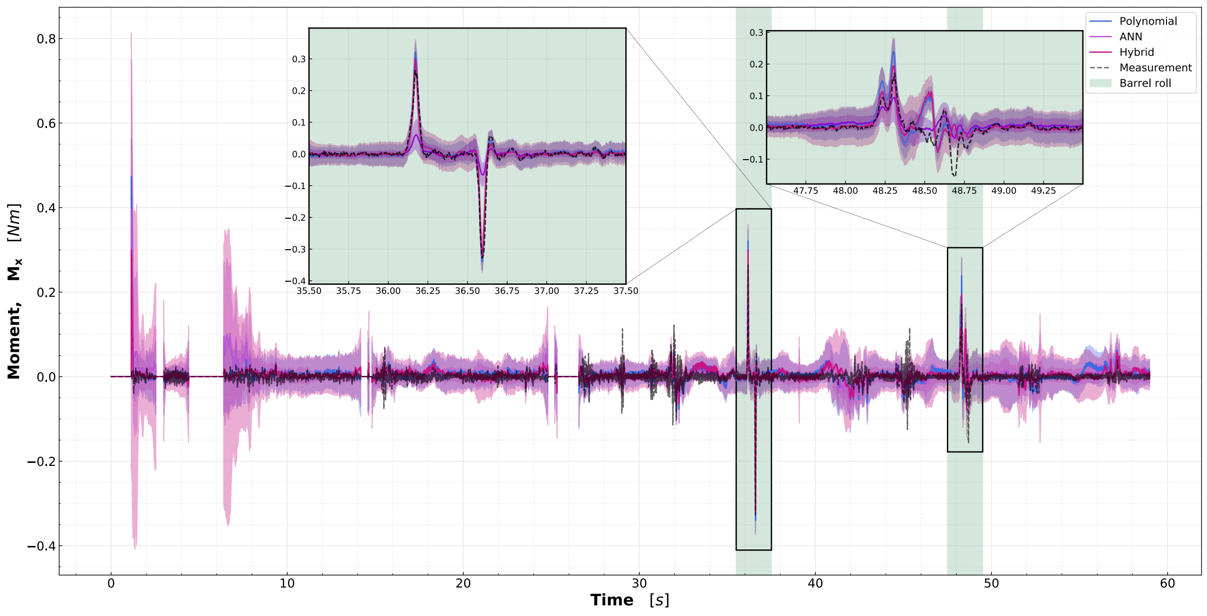


Fig. 24 Demonstration of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_x for flight 37 of the MetalBeetle involving several aggressive manoeuvres. Also shown is the measured M_x response (black dotted line). This flight was a first-person-view outdoor flight in windy conditions (wind speed $\approx 6\ ms^{-1}$ with gusts of up to $\approx 10\ ms^{-1}$). Highlighted in green are manoeuvres corresponding to barrel rolls

was affecting the second barrel roll. In any case, the moment models - especially the polynomial-based - appear capable of capturing the moments involved with some of the aggressive manoeuvres despite unknown wind. This again shows promise for the identification and development of outdoor quadrotor models.

VI. Analysis of identified models

While the model predictive performances give insight into how well the models fit the data, they do not alone describe how realistic and useful the models actually are. It is therefore important to analyze the identified models further to investigate their stability and sensitivity to the inputs. Indeed, while a model may give a good fit, any instabilities therein limit its operating envelope and thus practicability. Moreover, the identified MetalBeetle models are also implemented in a quadrotor simulation to evaluate their responses to simple step inputs to the attitude. Their subsequent behavior should be feasible and realistic for them to be considered valid and useful. Consequently, the analysis of the identified models discussed here entails a description of the influential states which modulate a models' predictions and the associated stability of these terms - given in section VI.A - and finally a simulation of the models to attitude step inputs is used to validate them in section VI.B.

A. Identified model structures & sensitivities

One of the benefits of the polynomial models is that their model structures provide direct insight into which terms are influential to the model. Indeed, the selected regressors may be interpreted physically and may reflect some real phenomena at play. Moreover, the additive accuracy of these regressors give an indication of the relative importance that these regressors have for the model. While this input importance is more difficult to ascertain with the ANN models, the final weights associated with the inputs of the model provide insight into which of these are influential states [19]. However, unlike the polynomial models, the interaction effects between these states captured by the ANN models is much more difficult to determine.

Similarly, the model stability of the black-box ANN models are difficult to quantify while the covariances of regressor coefficients of the polynomial models are a direct indication of their stability. For polynomial models, small regressor covariances indicate minimal variation of the associated regressor's coefficient value, and thus imply a stable and reliable model. In this paper, covariances are presented as a percentage of the associated regressor's coefficient magnitude to more easily compare the stability between regressors. A parallel may be drawn for the ANN models by investigating the variance in a given input-weight pair over the final training epochs. Analogous to the polynomial covariances, it is expected that a low weight variance implies a more stable model than those with high variances. Again, for inter-weight comparisons, these variances are expressed as a percentage of their associated weight magnitude. Through these simple analysis methods, the stability and reasonability of the influential model input states may be evaluated for all of the identified MetalBeetle models.

1. Polynomial model of the MetalBeetle

The identified MetalBeetle outdoor polynomial model for F_x harbors regressor instabilities and exhibits clear contamination by wind. The associated regressors may be found in table 5 in section VIII.A.5. Instabilities are immediately apparent upon the addition of the first regressor, $\cos(\theta)\bar{\omega}_{tot}^4$, with a high covariance of 36.6% of the coefficients magnitude. This high covariance, along with curious association of $\bar{\omega}_{tot}$ with $\cos(\theta)$ and not $\sin(\theta)$, suggests that this regressor attempts to capture the component of the forces along F_x produced by the total rotor speed, and interaction effects therein, but is mislead by the contamination of wind in the measurement data. Therefore, it likely fails to adequately model the intended phenomena. The influence of wind on the model is also apparent through the inclusion of $\sin(\psi)\mu_{v_{in}}\bar{\omega}_{tot}^3$, through the association with the yaw angle, ψ. Indeed, a reliance on ψ likely derives from the offset of the quadrotor's x-axis with the direction of incident wind to account for the unknown airspeed. Note that the interaction of $\mu_{v_{in}}\bar{\omega}_{tot}$ represents the effects of the propulsion system and the presence of the advance ratio of the induced velocity, $\mu_{v_{in}}$, may also encapsulate some thrust variance and blade flapping effects. Moreover, since this regressor is the only one which explicitly includes a dependence on the wind direction, it may be a prominent and significant effect. Indeed, other selected regressors, such as $\sin(\theta)\mu_{v_{in}}\bar{\omega}_{tot}$, also include this interaction. Hence, some underlying dynamics may nonetheless be recognized by the polynomial F_x model despite the prevalent wind. However, the high covariance of the first selected regressor, and clear detrimental effects of wind, imply that the outdoor model is invalid and unreliable.

Similarly, wind contamination incites instabilities for the outdoor MetalBeetle polynomial model of F_y. Despite this, some reasonable regressors constitute the final polynomial model. These regressors are summarized in table 6 in section VIII.A.5. The first selected regressor, $\sin(\phi)\bar{\omega}_{tot}$, is reminiscent of the first regressor of the F_x model but with a low covariance (0.01% of the associated coefficient's magnitude) and expected dependence on $\sin(\phi)$. As such, the outdoor F_y model likely represents some of the underlying dynamics despite the incursion of unknown wind. However, the effects of this wind are nevertheless evident in the model through regressors such as $\cos(\psi)\mu_{v_{in}}^2\bar{\omega}_{tot}^2$ which explicitly include the yaw angle. Again, parallels may be drawn with respect to the outdoor F_x model, which elects a similar term associated with the same states but modulated by $\sin(\psi)$. Hence, this lends further support to the hypothesis that the interaction of $\mu_{v_{in}}$ and $\bar{\omega}_{tot}$ is integral to describing the aerodynamic forces, and perhaps, is descriptive of high-speed flight. Indeed, the addition of this regressor boosts the accuracy of the outdoor model from $R^2 = 0.3453$ to $R^2 = 0.4009$. Nonetheless, the inclusion of the yaw angle results in an invalid model. This model invalidity is further supported by model instabilities from high covariances for some of the selected regressors. For instance, the forth regressor, $\bar{\omega}_{tot}\bar{p}^3$, hosts an unacceptable covariance of 190% of its associated coefficient magnitude.

In stark contrast to the outdoor MetalBeetle polynomial F_x and F_y polynomial models, the F_z model boasts stability and elects mostly relevant regressors. However, the effects of wind still corrupt the regressor choices of the outdoor F_z model. The associated regressors are summarized in table 7 in the appendix (section VIII.A.5). Given the extensive development of analytical thrust models in literature, it is perhaps unsurprising that the fixed regressors alone already flaunt decent model accuracy ($R^2 = 0.6210$). Many of the subsequently selected regressors are reasonably associated with the total rotor speeds, such as $\bar{\omega}_{tot}^4$. Moreover, despite the presence of wind, there are traces of symmetries between selected regressors. For instance, both $\cos(\phi)\mu_{v_{in}}\bar{\omega}_{tot}$ and $\cos(\theta)\bar{v}_{in}\bar{\omega}_{tot}^3$ are selected as regressors and, while different in composition, are likely intended to capture the component of thrust that contributes to forces along z while in motion along x and y. These terms probably differ due to the effects of wind and over-fitting. Their appearance in the model nonetheless accentuates the importance such symmetrical effects. The successive addition of regressors steadily improves the model accuracy implying that no one regressor dominates the underlying dynamics. The associated covariances of the regressors do not exceed 0.5% and therefore imply a stable model.

Likewise, the outdoor MetalBeetle polynomial model of M_x selects reasonable regressors that are stable. The selected regressors themselves may be found in table 8. Intuitively, the control rolling moment, $\bar{u}_p$, total rotor speed,

$\bar{\omega}_{tot}$, form components of many of the selected regressors. Examples include $\bar{\omega}_{tot}\bar{u}_p$, $\bar{\omega}_{tot}\bar{p}$, and $\bar{\omega}_{tot}\bar{u}_p|\bar{u}_r|$, which advocates for their necessity in describing the underlying dynamics. This is especially true for the first selected regressor, $\bar{\omega}_{tot}\bar{u}_p$, which promotes a significant jump in accuracy from $R^2 = 0.3092$ to $R^2 = 0.5848$ for the training data subset. Nonetheless, the effects of wind are apparent in the outdoor model due to its dependence on the yaw angle, through $\cos(\psi)\bar{\omega}_{tot}^2$, which should otherwise not influence the model. Despite the contamination by wind, the selected regressors for outdoor models show extremely low covariances ($< 0.01\%$ of the associated coefficient's magnitude). Thus, the identified outdoor polynomial M_x MetalBeetle model is likely stable but invalid due to the wind contamination.

In contrast, the outdoor pitching moment, M_y, polynomial model of the MetalBeetle harbors regressors with instabilities. The associated regressors may be found in table 9 and involve both reasonable and dubious selections. Analogous to the M_x models, many of the selected regressors are constructed with the control pitching moment, $\bar{u}_q$, and the total rotor speed, $\bar{\omega}_{tot}$. Again, both the indoor and outdoor models select $\bar{\omega}_{tot}\bar{u}_q$ as their first regressor, accentuating its importance in capturing the underlying dynamics. Recall that $\bar{\omega}_{tot}\bar{u}_p$ was also selected first for both M_x models. This strongly suggests that the linear interaction between $\bar{\omega}_{tot}$ and the corresponding control moment is descriptive of some of the moment dynamics outdoors. However, the M_y model elects terms associated with the yaw angle, such as $\cos(\psi)\bar{\omega}_{tot}^3$, which should be inconsequential to the pitching moment model but appear due to the contamination of unknown wind. Moreover, the outdoor M_y model fosters an unstable regressor, $\bar{\omega}_{tot}\bar{q}^3|\bar{r}|$, with a high covariance (19.7% of the associated coefficient's magnitude), added towards the end of the step-wise selection procedure. Subsequently, the identified outdoor M_y model may be considered unstable and unreliable.

The outdoor polynomial model for the yawing moment, M_z, elects both regressors that correlate well with the underlying dynamics and some that are questionable. These regressors summarized in table 10. Peculiarly, the outdoor yawing moment erroneously favors terms related to the control rolling moment, $\bar{u}_p$. While the exact reason for this curious preference is unknown, it is likely a symptom of the prevalent wind or over-fitting. Nonetheless, the conspicuous linear interaction of $\bar{\omega}_{tot}\bar{u}_r$ is again apparent in the yawing moment model. This lends support to the hypothesis of this interaction being integral to describing all the moment dynamics. Moreover, a higher order variant, $\bar{\omega}_{tot}\bar{u}_r^3$, is also selected by the M_z model. Consistent with all the identified outdoor models, the influence of the unknown wind in the outdoor model is established through a yaw angle dependency in $\sin(\psi)|\bar{u}_q||\bar{u}_p|^3$. Moreover, the outdoor M_z model maintains stability with low covariances for all regressors ($< 0.06\%$). Hence, M_z model may be considered stable, but likely produces invalid results due to wind contamination.

2. *Hybrid model of the MetalBeetle*

The goal of the hybrid models is to augment the underlying polynomial predictions in an effort to improve accuracy. To get a complete overview of the stability and sensitivities of the hybrid models, the ANN components may be scrutinized. Accordingly, the weight magnitudes of the input layer, averaged over the last 10% of training epochs (i.e. 15 epochs), provides insight into any significant inputs recognized by the ANN compensators. The stability of the ANN compensator is interpreted through the associated variability of these weights. For the hybrid force models, these weight magnitudes and variances are respectively illustrated in fig. 25 and fig. 26. Likewise, fig. 27 and fig. 28 depicts those for the hybrid moment models.

The outdoor hybrid model of F_x exhibits clear dependency on the total rotor speed but suffers from consistent instabilities, particularly with respect to the control moments. Indeed, the underlying polynomial model recognizes the importance of $\bar{\omega}_{tot}$ but struggles to incorporate its effects due to the contamination of wind. Consequently, the high weight magnitude of $\bar{\omega}_{tot}$ in the hybrid model may seek to remedy this. Given the prominence of this input, it may also be the case that there are additional rotor interaction based effects which materialize in the outdoor flight regime. Other inputs, although ancillary to $\bar{\omega}_{tot}$, involve the x and y component of the advance ratios μ_x and μ_y respectively. The apparent relevance of μ_y likely arises from the effects of wind. Likewise, the scattered importance of the attitude angles - aside from those related to θ - may also surface from wind effects and thus facilitate over-fitting. These dependencies therefore suggest an unreliable model. Arguments of an unreliable hybrid model are also supported by instabilities present in the ANN compensator. These instabilities appear to be concentrated on the control moments $\bar{u}_p$, $\bar{u}_q$, and $\bar{u}_r$ and may relate to the adjustments made by the controller to reject disturbances during flight. Perhaps there are also some dynamics here which the hybrid model is struggling to fit. While other variances in the model are mostly low and imply stability, the consistent instability of the control moments is a cause for concern and questions the reliability and validity of the outdoor MetalBeetle hybrid model of F_x.

Contrarily, the outdoor MetalBeetle hybrid model for F_y exhibits a more tranquil ANN compensator with better model stabilities and more subtle weight magnitudes. These almost uniform weight magnitudes imply that the ANN

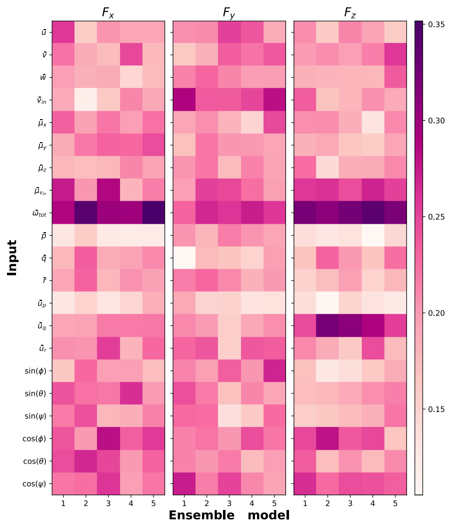


Fig. 25 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the ANN compensator component of the identified hybrid force models of the MetalBeetle (*Outdoors*).

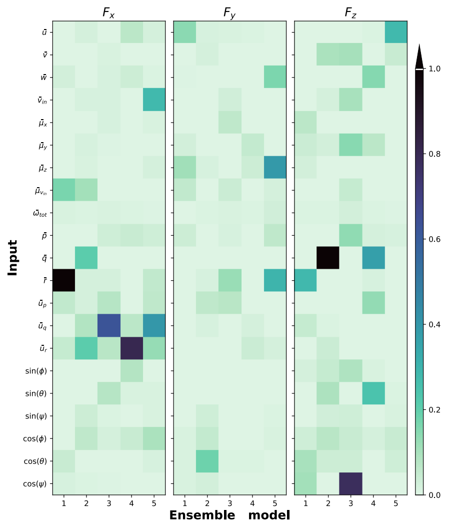


Fig. 26 Variance in input layer weights, taken over the last 10% of training epochs, of the ANN compenstor component of the identified hybrid force models of the MetalBeetle (*Outdoors*).

model struggles to identify useful dynamics in the error residual of the underlying polynomial. The most notable weights are associated with ω_{tot} and $\bar{v}_{in}$. Collectively, these terms relate to the propulsion system of the quadrotor and, thus, the improvements seen in model performance over the underlying polynomial likely stem from phenomena related to the rotors and interaction effects therein. These may potentially be associated with the effects of thrust variance given the reliance on $\bar{v}_{in}$. The scattered importance of the attitude angles implies over-fitting and the relatively high magnitudes for the yaw angle, ψ, likely reflect the effects of wind. Despite this, the ANN compensator for the hybrid model of F_y enjoys generally low variance weights. Therefore, the ANN compensator may be considered stable, although perhaps not valid.

The outdoor MetalBeetle hybrid model of F_z appears to capture additional phenomena that the underlying polynomial model is oblivious to given the defined input significance. However, the associated variances in weights imply some model instability. The ANN compensator attributes high magnitudes to $\bar{\omega}_{tot}$ and $\mu_{v_{in}}$ to encapsulate effects linked to the rotor speeds and interactions therein. These inputs almost coincide exactly with the fundamental terms of the analytical models of thrust variance and blade flapping. However, some of these interactions may involve the wind, given the consistent and moderate weight magnitudes of $\cos(\psi)$. Rather unexpectedly, the ANN compensator also considers the control pitching moment, $\bar{u}_q$, a significant input. While the control moments are speculated to be relevant to the dynamics of F_z, the lonesome importance of $\bar{u}_q$ is perhaps more indicative of over-fitting. A potential explanation for its prevalence, however, is the preponderance of FPV flights in the identification data which inherently involves more pitch-based excitation. The majority of the weight variances of the ANN compensator are low, however, there are a few scattered high variance terms which are a cause for concern. Of note are the instabilities associated with the pitch rate, $\bar{q}$, and yaw angle through $\cos(\psi)$. Therefore, although reasonable dynamics may be recognized, predictions made by the hybrid F_z model should be interpreted with caution as they may be unstable.

In contrast to the force hybrid models, and rather unexpectedly, the ANN compensators display worse performance for the outdoor models in comparison to their underlying polynomials. This is likely a result of inconsistent weight magnitudes among the constituent ANN ensembles (refer to fig. 27).

This is evident for the identified outdoor M_x hybrid model, which struggles to isolate influential inputs and harbors some concerning model instabilities. While most constituent ANNs in the ensemble of the outdoor hybrid M_x model reasonably afford modelling importance to $\bar{\omega}_{tot}$ and $\bar{u}_p$, not all do. Take for instance, the lack of significance of these terms for the second ANN in the ensemble. It may be argued that this ANN has not converged, but the variability of weight magnitudes among the remaining inputs is uncharacteristic of such non-convergence. The inability of the second ANN to recognize to the emergent dynamics reflects the challenges of the modelling task and the risk of over-fitting in

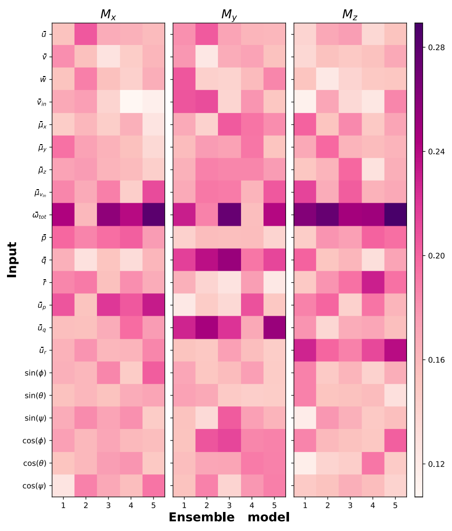


Fig. 27 **Input layer weight magnitudes, averaged over the last 10% of training epochs, of the ANN compensator component of the identified hybrid moment models of the MetalBeetle (*Outdoors*).**

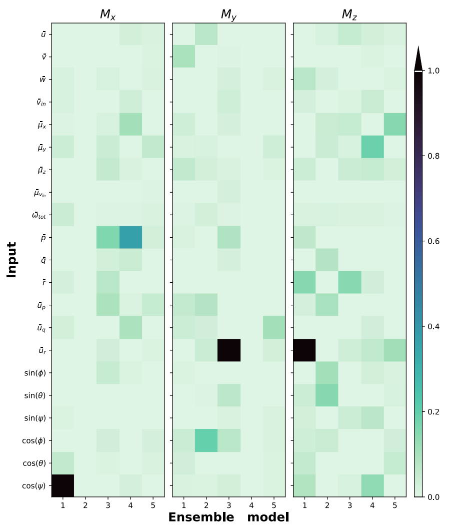


Fig. 28 **Variance in input layer weights, taken over the last 10% of training epochs, of the ANN compenstor component of the identified hybrid moment models of the MetalBeetle (*Outdoors*).**

such a hybrid approach. Rotor speed based effects appear to be important for describing the dynamics in the polynomial error residuals, as $\bar{\omega}_{tot}$ hosts the highest weight magnitudes for the remaining ANNs. The contamination of wind is evident through the high variance in input weights associated with $\cos(\psi)$ for the first ANN. Moreover, there is some high input variance consistency across some of the constituent ANNs associated with the roll rate, $\bar{p}$, which is undesirable as it modulates some of the rolling moment dynamics. Through these high weight variances, and the generally incoherent weight magnitudes among its constituent ANNs, the hybrid outdoor M_x model is likely unstable and invalid. This potentially explains some of the deterioration in performance experienced by this model over the underlying polynomial.

Similarly, the outdoor hybrid model of the MetalBeetle pitching moment, M_y, suffers from inconsistent weight magnitudes with some high variance weights. Such inconsistencies are apparent through the scattered relevance of $\bar{\omega}_{tot}$, $\bar{u}_q$, and $\bar{q}$ across the constituent ANNs. While, collectively, these inputs modulate some of the dynamics behind M_y, the lack of consistency is indicative of the ANN compensator struggling to extract emergent dynamics from the polynomial model error residuals. Again, the offending ANN models may not be discounted as unconverged as they peculiarly allocate high magnitudes to other inputs, such as $\bar{u}_p$. Hence, these constituent ANNs appear to seek inputs that over-fit the training data. Indeed, this may also be a byproduct of the unknown wind contamination in the training data, which complicates the modelling task. In any case, the outdoor hybrid model of M_y may be considered unreliable and invalid. While this model generally experiences low variances, there is a concerning high variance weight associated with $\bar{u}_r$ which may lead to instabilities due to the dense construction of the ANNs.

In contrast, the outdoor hybrid yawing moment, M_z, model of the MetalBeetle consistency associates high magnitudes with justifiable inputs. However, some of the present instabilities are consequential for the outdoor models' reliability. All constituent ANNs agree on the significance of $\bar{\omega}_{tot}$ and the control yawing moment, $\bar{u}_r$, which understandably influence the yawing moment. While the underlying polynomial also allocates importance to these terms, it may lack the approximation power to capture some of the prevalent dynamics. Most of the associated weight variances are generally low, with the exception of one notably unstable neuron. Unfortunately, this instability is associated with $\bar{u}_r$ for the first ANN in the ensemble. As $\bar{u}_r$ is a significant modelling term, this instability is alarming and probably leads to poor and unreliable predictions. As such, the outputs of the outdoor hybrid M_z model should be interpreted with caution as they are likely unreliable due to concerning instabilities.

3. ANN model of the MetalBeetle

The weight magnitudes, averaged over the last 10% of training epochs, of the identified ANN force models of the MetalBeetle and the variances therein are illustrated in fig. 29 and fig. 30 respectively.

The identified outdoor MetalBeetle ANN F_x model allocates the highest weight magnitudes to the total rotor speed, $\bar{\omega}_{tot}$, advance ratio of the induced velocity, $\mu_{v_{in}}$, and, to a lesser extent, the pitch angle through $\sin(\theta)$. Again, these inputs are associated with the effects of blade flapping and thrust variance discussed for the analytical models of the quadrotor. These inputs are in agreement with the equivalent polynomial model for which these states form integral regressors. Therefore, the ANN may recognize the effects of the rotor system on F_x as modulated by the pitch angle. Moreover, the outdoor model appears to allocate some importance the control pitching moment, $\bar{u}_q$, and the yaw angle, through $\cos(\psi)$. While $\bar{u}_q$ may indeed be related to some underlying dynamics, the yaw angle is reflective of the effects of wind. Such an influence of the yaw angle is also apparent in the corresponding polynomial model. The outdoor ANN F_x model enjoys mostly low variance weights, which imply model stability. However, there may be some consistency in relatively high model variances associated with the attitude angles (e.g. $\sin(\phi)$) which is a cause for concern. While this may be a byproduct of the wind contamination, it may nonetheless lead to model instabilities.

In contrast, input importance is much less clear for the outdoor ANN F_y model, which is perhaps reflective of the difficulty in modelling task and contamination of wind. Nonetheless, the importance of ω_{tot} and $\mu_{v_{in}}$ is again recognized, which implies that at least some fundamental dynamics have been identified. However, the lack of significance allocated to inputs expected to be related to the F_y (e.g. $\sin(\phi)$) casts doubt on the validity of the outdoor F_y ANN model. Recall that such inputs are seen in the polynomial models. Nonetheless, the generally low weight variances and lack of consistently in high variance-input pairs suggests model stability. However, there remain some high variance weights ($\approx 50\%$) which may propagate instability throughout the model, hence predictions should be interpreted with caution.

Contrary to the outdoor MetalBeetle ANN F_x and F_y models, the outdoor ANN F_z model shows a clearer dependence on inputs through established high weight magnitude terms. Rather intuitively, these are allocated to $\bar{\omega}_{tot}$ and $\mu_{v_{in}}$. These states are associated with the thrust variance and blade flapping effects and correspond well with the equivalent polynomial model. The outdoor F_z ANN model also recognizes some importance in $\bar{u}_q$ and $\cos(\psi)$. As discussed for the hybrid F_z model, the prominence of $\bar{u}_q$ may be a byproduct of the large number of FPV flights conducted outdoors, or the presence of wind, or both. Likewise, the relative importance of $\cos(\psi)$ probably stems from the effects

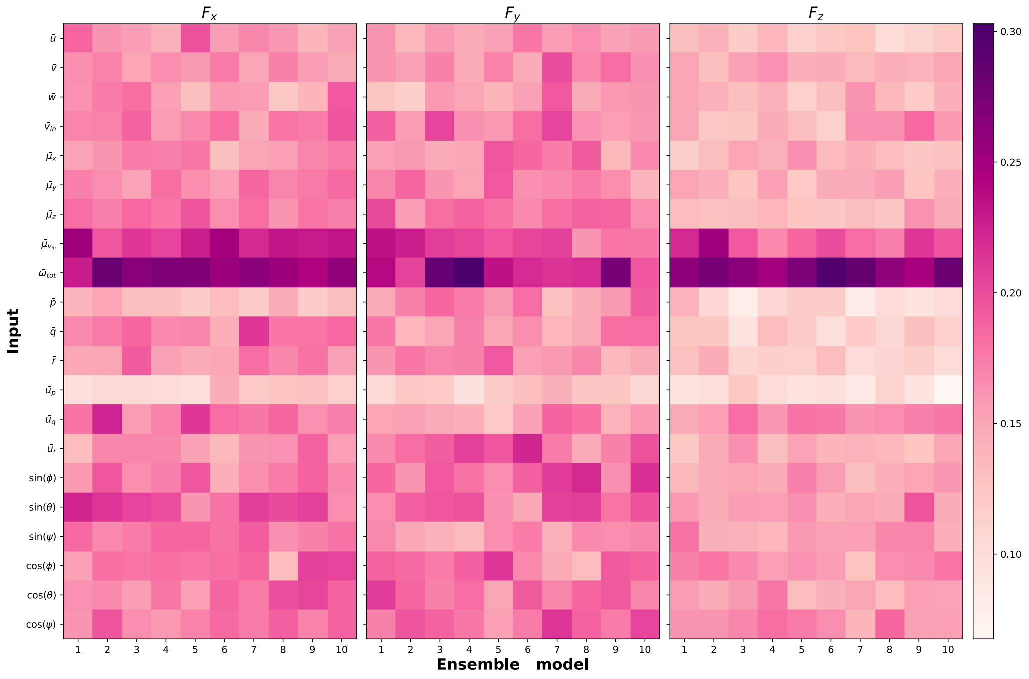


Fig. 29 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the identified ANN force models of the MetalBeetle (*Outdoors*).

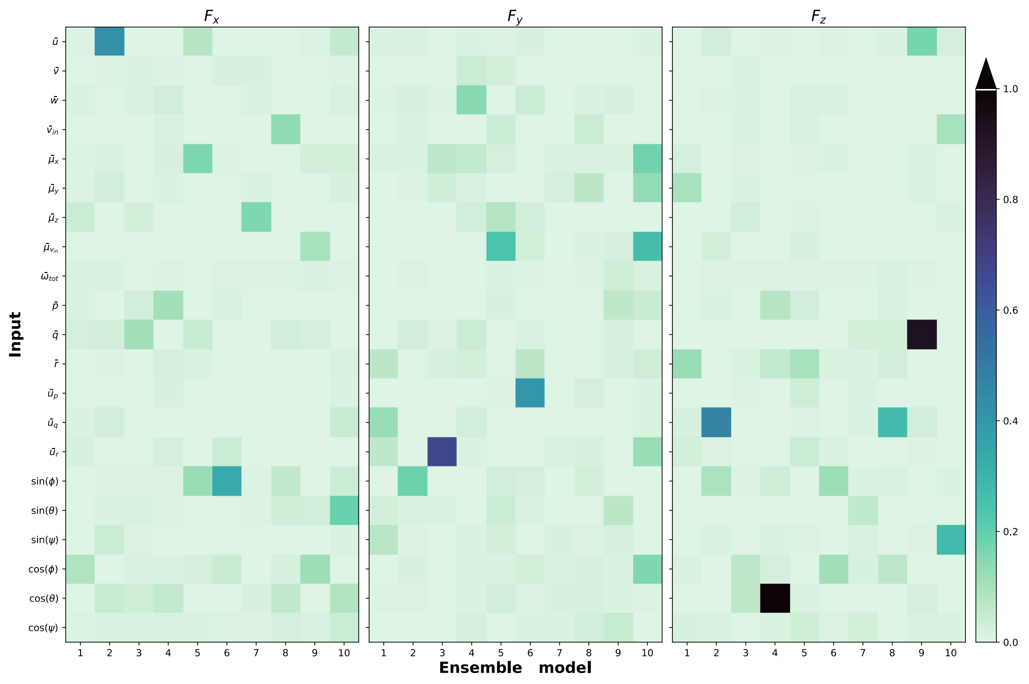


Fig. 30 Variance in input layer weights, taken over the last 10% of training epochs, of the identified ANN force models of the MetalBeetle (*Outdoors*).

of wind alone. While the model variances are generally low, there are a few concerning weights with unacceptably high variances (i.e. $\approx$ 100% of the corresponding weight's magnitude). Although these are associated with low weight magnitude terms, the instabilities may nonetheless propagate throughout the ANN model due to the dense construction.

The principal observations made for the hybrid outdoor moment models of the MetalBeetle are transferable to the ANN-only models. In particular, the outdoor ANN models struggle to isolate significant inputs due to inconsistent weight magnitudes across the ANN ensembles. These input weight magnitudes, averaged over the last 10% of training epochs (i.e. 15 epochs), are depicted in fig. 31. The accompanying weight variances are illustrated in fig. 32.

Indeed, the identified outdoor ANN rolling moment, M_x, model of the MetalBeetle finds difficulty in describing any coherent dynamics but sports generally low associated weight variances. Only shadows of the expected significant inputs - such as $\mu_{v_{in}}$, $\bar{\omega}_{tot}$, and $\bar{u}_p$ - are visible in the outdoor ANN M_x model through scattered weight magnitudes across the constituent ANNs for these inputs. This implies that the outdoor ANN model likely recognizes some of the underlying dynamics but struggles to formalize these while getting distracted by irrelevant effects or noise. This perhaps explains its poor performance with respect to the outdoor M_x polynomial model. This inability to learn effectively is also reflected by the relatively low weight variances of the outdoor ANN model, which imply stability (i.e. convergence).

In comparison, the identified ANN outdoor pitching moment, M_y, model of the MetalBeetle shows a clearer reliance on some of the expected inputs but exhibits more instabilities than the M_x model. Reasonably, the control pitching moment, $\bar{u}_q$, emerges as the most important input followed by the the pitch rate, $\bar{q}$, across all constituent ANNs. This implies that the ANN M_y model is able to recognize some of the fundamental dynamics. Moreover, the associated variances are generally low, suggesting decent model stability and reliability. However, due to the presence of some relatively high variances, the outdoor ANN M_y results should be interpreted with caution.

Contrarily, the identified outdoor ANN yawing moment, M_z, model of the MetalBeetle exhibits the highest instabilities of the moment models and expresses anxiety in selecting significant inputs. Although, emergent inputs are still visible through the yawing control moment, $\bar{u}_r$, the total rotor speed, $\bar{\omega}_{tot}$, and advance ratio of the induced velocity, $\mu_{v_{in}}$. These inputs correlate well with physical phenomena and the equivalent polynomial model. However, there appears to be consistently high variances associated with $\sin(\theta)$ which may induce some unpredictable outputs, especially given the prevalence changes in pitch during FPV flight. This implies an invalid outdoor ANN M_z model.

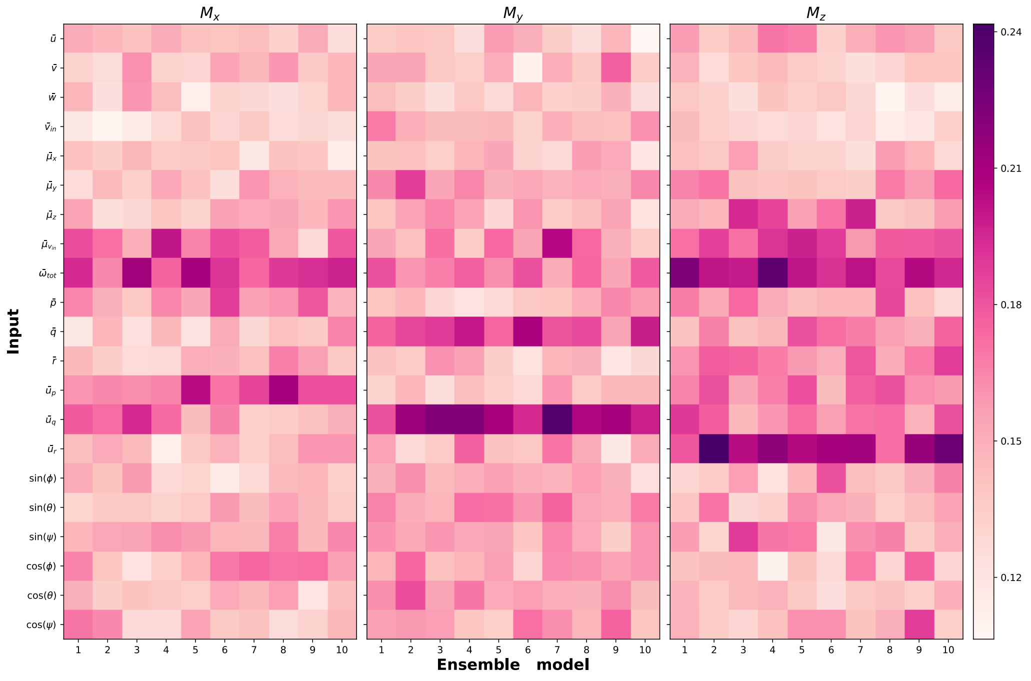


Fig. 31 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the identified ANN moment models of the MetalBeetle (*Outdoors*).

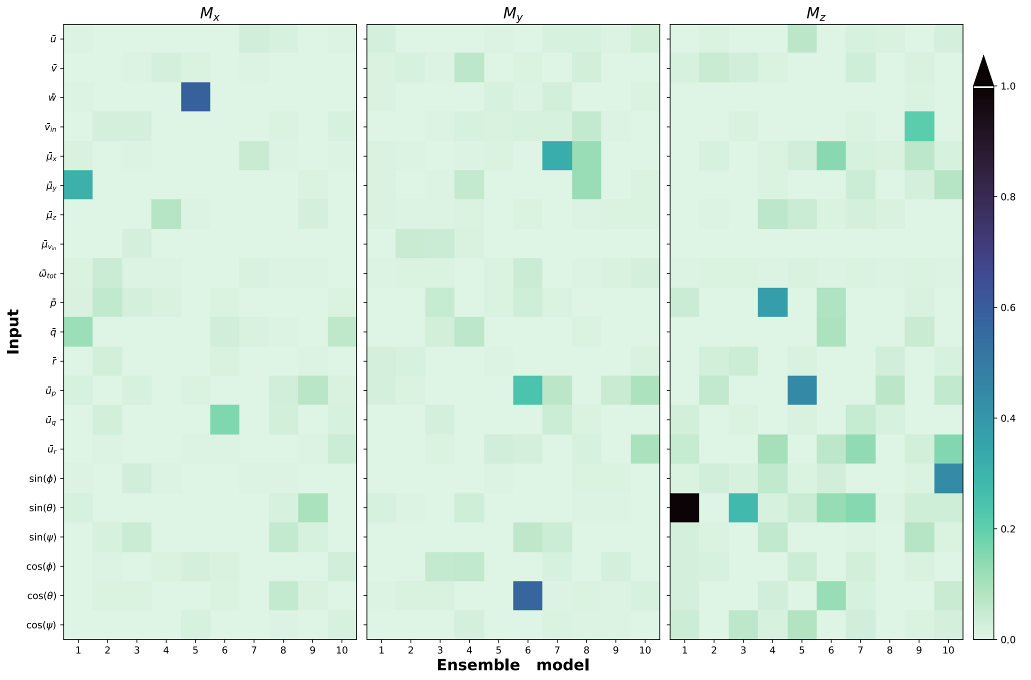


Fig. 32 Variance in input layer weights, taken over the last 10% of training epochs, of the identified ANN moment models of the MetalBeetle (*Outdoors*).

B. Simulation of identified models

Though the performance metrics may indicate that models perform well, or poorly, it is difficult to abstract this to determine how useful the models actually are. Indeed, good performance is a promising start for a realistic model, however, these metrics only evaluate such performance with respect to the measurement data. Therefore, in order to verify if the identified models are indeed useful, a simple quadrotor simulation is developed. In this simulation, the quadrotors are tasked with following fundamental attitude references, such as a step input to the pitch angle. Their subsequent responses to these reference inputs may be evaluated for feasibility. Indeed, one of the main motivators behind developing better quadrotor models is to be able to use them in simulation. As such, the quadrotor models identified in this paper are ideally suitable for this purpose. To contextualize the results of the simulation, a brief overview of the simulation platform is given prior to presenting the associated results.

The quadrotors (i.e. models) in the simulation are controlled via attitude, rate, and vertical velocity PID controllers. Note that linear controllers, such as PIDs, are actually unsuitable for controlling a non-linear system like the quadrotor across its entire flight envelope. Paradoxically, PIDs are exclusively employed by BetaFlight to facilitate the control of the quadrotor. In BetaFlight, the non-linearities are circumvented through a form of adaptive control - effectively localizing linear control around different parts of the flight envelope. Such control is beyond the scope of this research and thus only a simple PID attitude, rate, and vertical velocity controllers are employed here. An illustration of the control loop and simulation structure used here is given in fig. 33.

Briefly, the simulation is composed of an inner loop which controls the rotational rate of the quadrotor. Here, a rotational rate command is transformed into a change in rotor speeds. The rotational rate command is obtained through the attitude controller in the outer loop. The velocity controller is in a separate loop and controls the thrust of the quadrotor to maintain altitude in the earth frame. Moreover, rotor actuation dynamics and saturation limits are imposed for the simulation with parameters taken from the simulation of Sun et al. [21]. The rotor speeds are saturated to the limits of the quadrotor used for model identification while the rotor dynamics restrict the rotational acceleration.

The initial state - composed of the attitude, velocity, and rotational rate - along with an initial rotor speed command form the inputs to the simulation. At each simulation step, the current states, and aggregates thereof, are inputted into the identified aerodynamic models to obtain the forces and moments. These are subsequently fed through the equations of motion (i.e. eq. (3) and eq. (2)) to derive the state derivative, which is used to obtain the subsequent state (through

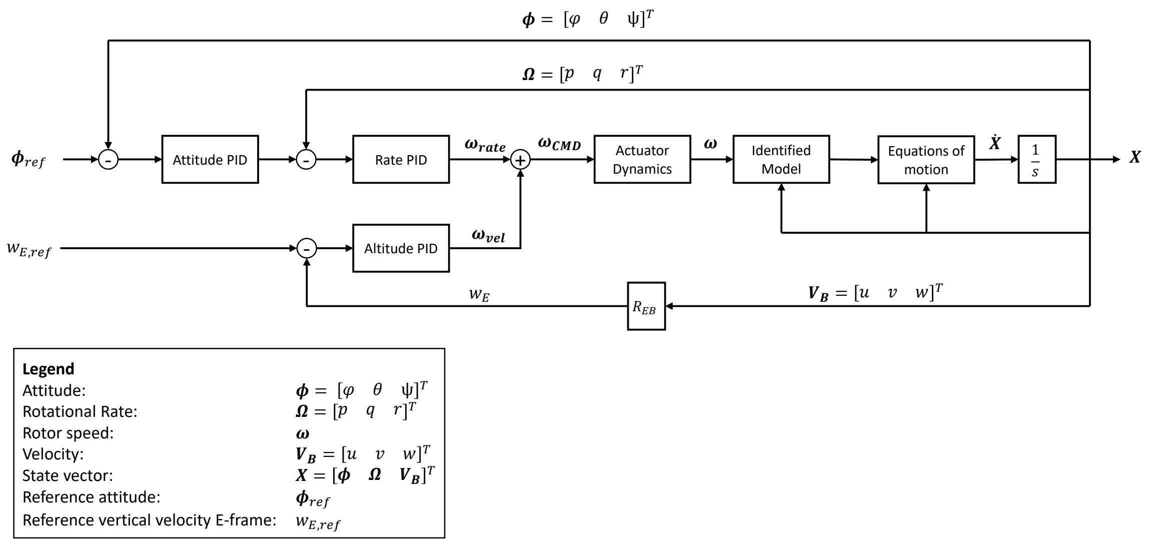

$\boldsymbol{\Phi} = [\varphi \quad \theta \quad \psi]^T$
$\boldsymbol{\Omega} = [p \quad q \quad r]^T$
$\boldsymbol{\Phi}_{ref}$
Attitude PID
Rate PID
$\boldsymbol{\omega}_{rate}$
$\boldsymbol{\omega}_{CMD}$
Actuator Dynamics
$\boldsymbol{\omega}$
Identified Model
Equations of motion
$\dot{\boldsymbol{X}}$
$\frac{1}{s}$
$\boldsymbol{X}$
$w_{E,ref}$
Altitude PID
$\boldsymbol{\omega}_{vel}$
w_E
R_{EB}
$\boldsymbol{V_B} = [u \quad v \quad w]^T$
Legend
Attitude: $\boldsymbol{\Phi} = [\varphi \quad \theta \quad \psi]^T$
Rotational Rate: $\boldsymbol{\Omega} = [p \quad q \quad r]^T$
Rotor speed: $\boldsymbol{\omega}$
Velocity: $\boldsymbol{V_B} = [u \quad v \quad w]^T$
State vector: $\boldsymbol{X} = [\boldsymbol{\Phi} \quad \boldsymbol{\Omega} \quad \boldsymbol{V_B}]^T$
Reference attitude: $\boldsymbol{\Phi}_{ref}$
Reference vertical velocity E-frame: $w_{E,ref}$

Fig. 33 Simulation architecture depicting the role of the PID controllers and incorporation of the identified models.

numerical integration; specifically the explicit Euler Method[***]). The simulation runs at 100 Hz and is programmed entirely in Python. It is important to note that there is a discrepancy between the body reference frame described in section II (i.e. fig. 1) and the body reference frame used for the simulation. The identified models, and thus simulation, follow the body reference frame of BetaFlight for simplicity in processing and interpretation (i.e. correlation with measurements and video logs) since all measured states are recorded with respect to this reference frame. In BetaFlight, the x-axis points forwards, y-axis to the left, and z-axis up[†††].

All of the identified quadrotor models are first subject to a simulated hovering flight. While such flight is perhaps outside the domain of validity for the models, it nonetheless provides valuable insight on their stability and inherent biases. Therefore, the detailed analysis of the hovering flight is given in the appendix (section VIII.B) with only a summary given for the corresponding models. The hovering simulation flight is also used to help tune and verify that the PID controllers are working as intended. Note that these controllers are largely the same across the polynomial-only, ANN-only, and hybrid models with some slight modifications to improve, for example, damping characteristics. The final PID values are summarized section VIII.C for each of the identified models. Moreover, the corresponding prediction intervals (PIs) are also illustrated in the simulation results for insight on the models' confidence.

1. Polynomial model of the MetalBeetle

The hovering simulation results of the outdoor polynomial model of the MetalBeetle, described in depth in section VIII.B.1, reveal the expected influence of wind. In particular, for $\psi = 0$, the F_x harbors a non-zero constant force (refer to fig. 77). This induces a runaway term in the corresponding velocity. Indeed, simulations of trimmed flights did not eliminate this constant F_x term, implying that it arises due to wind. It was found that this wind effect is mostly modulated by $\bar{\omega}^4_{tot}$ suggesting that the wind effects interact heavily with the rotor speed.

However, hovering flight is actually outside the domain of validity of the identified models since they are only identified on segments of the data wherein relevant excitations are present. Therefore, to more representative of this, the identified polynomial model's response oscillating attitude step inputs is simulated.

The corresponding commanded rotor speed and attitude responses to oscillating simultaneous step roll and pitch references are illustrated in fig. 34 and fig. 35 respectively. This particular input pattern is constructed to maintain control of the quadrotor and ensure that the velocities do not exceed the validity limits of the model. Despite the vigorous attitude changes (i.e. -0.8 to 0.8 rad), the outdoor MetalBeetle polynomial model is able to track the commanded

[***]Indeed, the forward Euler integration scheme can be notoriously unstable. However, it was found to be sufficient in this simulation and thus kept for simplicity.

[†††]Therefore, the BetaFlight axis system is still a right-handed system.

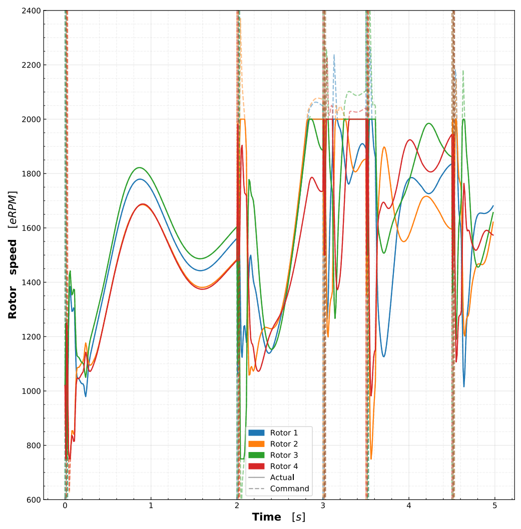


Fig. 34 Commanded rotor speed of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

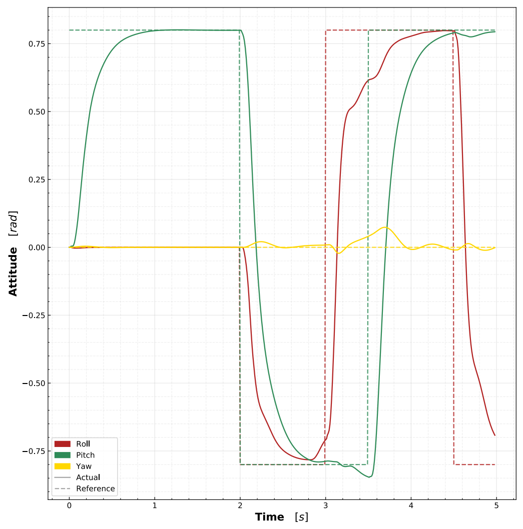


Fig. 35 Attitude response of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

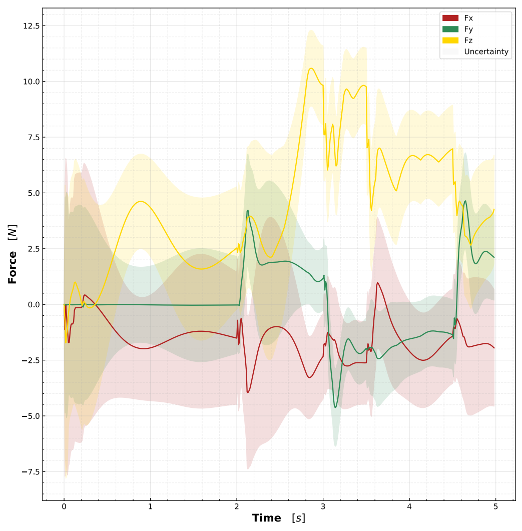


Fig. 36 Force response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

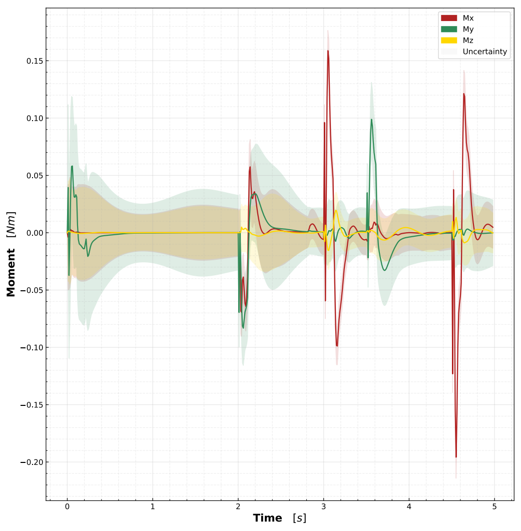


Fig. 37 Moment response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

attitude references satisfactorily. Indeed, these responses resemble the shapes of similar attitude commands seen in the measurement data (see, for example, pitch and roll attitudes in fig. 40 and fig. 41 respectively) and thus imply that the response dynamics and PID controllers are realistic. However, the attitudes occasionally struggle to meet their references, which is undesirable. This arises from the commanded rotor speeds exceeding the upper saturation limits of the rotors, as shown by the corresponding locations in fig. 34. Given the large rotations tasked of the quadrotor, the rotor speeds quickly rise to the saturation limits due to the sub-optimal PID controller. Aside from these controller induced rapid oscillations, the rotor speed responses also conform to physical expectations as the average rotor speed increases when the quadrotor has a non-zero attitude in order to maintain altitude.

The associated force response during these oscillating step roll and pitch commands is depicted fig. 36. The effects of wind are apparent in the simulated F_x response through the predominantly negative force. This is not reflective of the measured force for similar manoeuvres, as shown in fig. 40. Indeed, through the velocity response depicted in fig. 38, the apparent hesitation of the quadrotor to increase towards positive velocities is obvious. Note that, despite the mostly negative force, positive velocities arise from the effects of gravity as the quadrotor tilts. Since the regressor term, $\bar{\omega}_{tot}^4$, induces this negative bias for F_x (see hovering simulation results in section VIII.B.1), the magnitude of this negative force scales with the rotor speed. This explains the difficulty in increasing the velocity towards the positive. The negative association of $\bar{\omega}_{tot}^4$ likely stems from the (average) effects of the unknown wind. Physically, then, this response may be interpreted as flying into the wind for positive velocities and flying away from the wind for negative velocities. In contrast to the peculiarities of F_x, the F_y force model reflects reality well. In fact, the F_y force response is remarkably similar to the measurement data for similar attitude inputs, as shown in fig. 40 and fig. 41. Note, however, that in the identification data, the thrust was also increased following a direction change to expedite these changes and accelerate the quadrotor. This may explain some of the discrepancies between the simulation and reality. Likewise, the associated velocity responses in fig. 38 conform well with expectations. Note, however, that they do not reflect the outdoor measurement data well as seen in fig. 40 and fig. 41. This highlights the limitations of the outdoor velocity estimation through GPS. Indeed, the simulated velocity response more closely matches the measurements for the indoor experiments (see fig. 129 in section VIII.D). As such, future research should investigate ways to improve velocity estimations for outdoor data collection, including methods to directly measure the air speed. Similarly, the F_z response matches expectations wherein it increases following attitude changes in order to maintain altitude.

Furthermore, the simulated moment responses to oscillating roll and pitch step inputs are reasonable and reflect instances of the associated attitude transitions. The simulated moment responses are depicted in fig. 37. These moments are sharp and short in nature and resemble those found for similar inputs in the identification data, as illustrated by

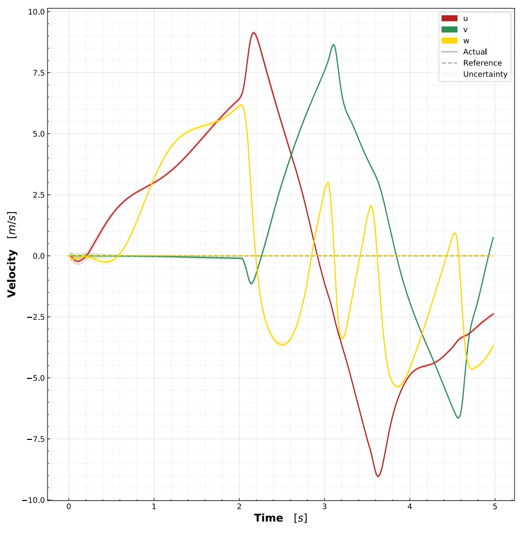


Fig. 38 Velocity response of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

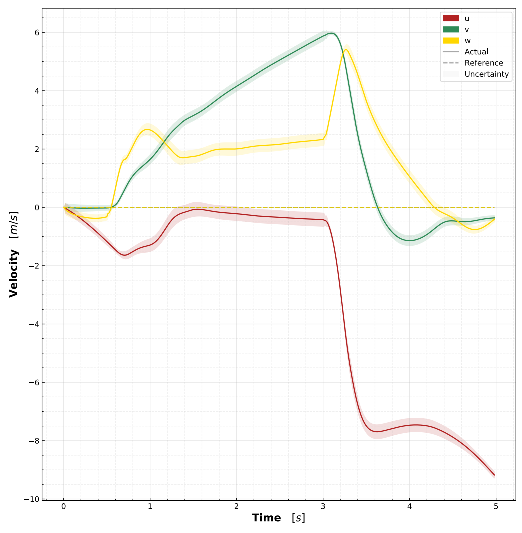


Fig. 39 Velocity response of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

fig. 42. Note that the rapid initial oscillations seen in the simulated moment response arise from the rapid oscillations of the PID controller. Such control inputs are not seen in the flight data, hence, this initial spike is due to the employed PID controller for simulation. Nonetheless, the subsequent response - for which the PID outputs behave - conforms to reality. Moreover, the associated PIs decrease in width for periods of greater excitation, reflecting the models' confidence with domains of the flight envelope similar to its identification set.

In order to evaluate excitations for the yawing moment, the response of the outdoor MetalBeetle polynomial models to oscillating yaw step inputs is also simulated. The associated attitude response is presented in fig. 44. This response demonstrates that the quadrotor is able to somewhat track the commanded yaw step inputs. There is a slight steady state

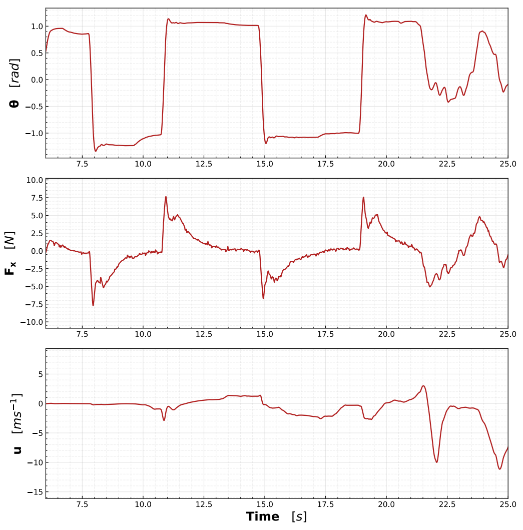


Fig. 40 Outdoor flight data of taken from the Metal-Beetle depicting the change in force, F_x, and velocity, u, due to oscillating commanded pitch angles, θ.

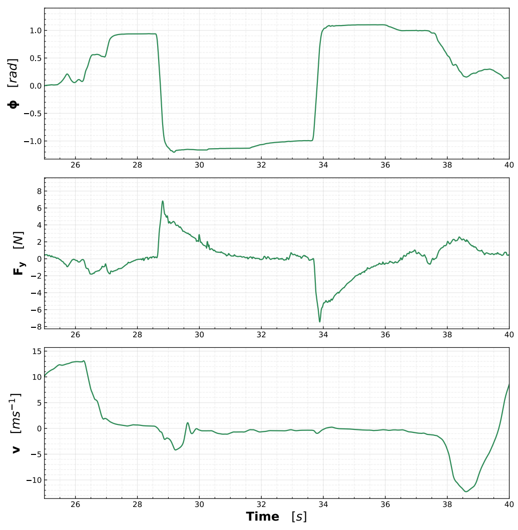


Fig. 41 Outdoor flight data of taken from the Metal-Beetle depicting the change in force, F_y, and velocity, v, due to oscillating commanded roll angles, ϕ.

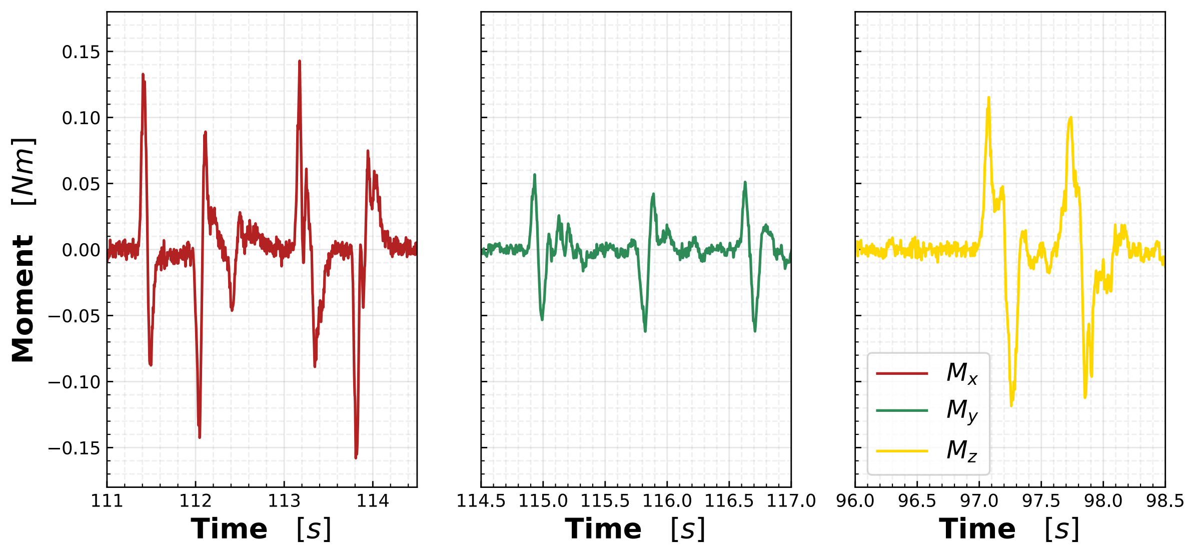
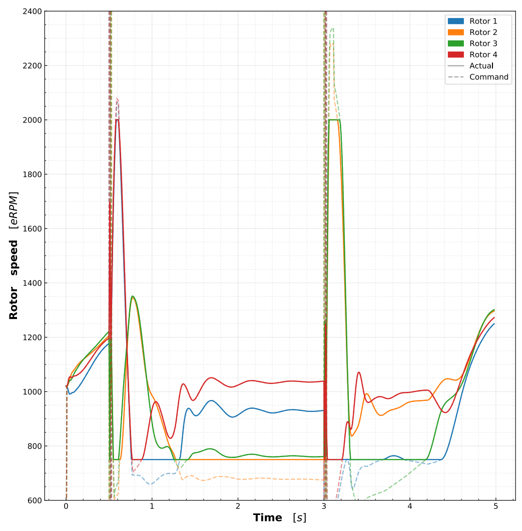


Fig. 42 Example of the measured moments during attitude step inputs in the identification data of the outdoor MetalBeetle.



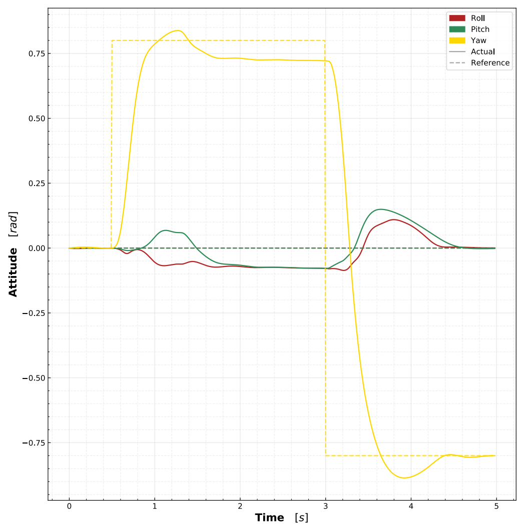


Fig. 43 Commanded rotor speed of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

Fig. 44 Attitude response of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

error in the first step input, and a non-zero mini-step in the roll and pitch angles, which is the result of the saturation of some of the rotors during this period as seen in fig. 43. This may be improved through a better altitude, or otherwise thrust, controller as the current implementation attempts to reduce the rotor speeds due to a net increase in thrust during the manoeuvre. This increase in thrust is induced by the rapid yaw rotation. To facilitate the yaw manoeuvre, rotors 1 and 4 shoot up to the upper saturation limit and rotors 2 and 3 to the lower. Given that the lower saturation bounds are closer to the hovering thrust than the upper bounds, the average thrust increases (i.e. the midpoint between the saturation limits is higher than the thrust needed to maintain hover). Indeed, after the final yaw manoeuvre, these rotor speeds return to idle as would be expected of the real quadrotor.

The corresponding force and moment responses for these yaw step inputs are illustrated in fig. 45 and fig. 46

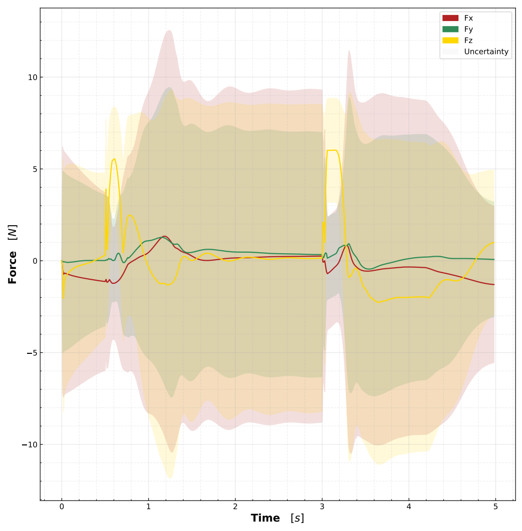


Fig. 45 Force response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

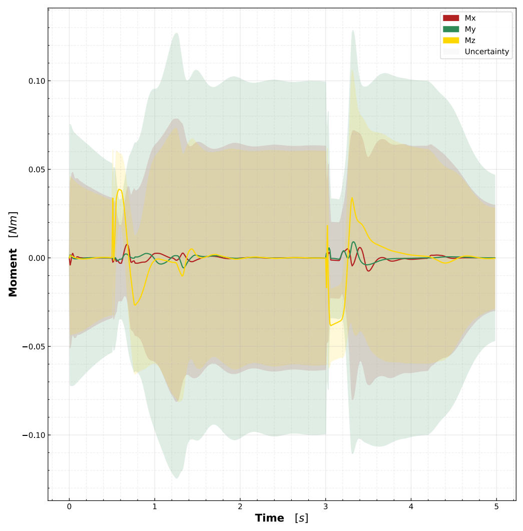


Fig. 46 Moment response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

respectively. The aforementioned yaw dependency of the force models is apparent in their simulated responses following a change in yaw angle. Indeed, there is a non-zero F_x before any yaw inputs (similar to that of the hovering simulation in section VIII.B). However, after the first yaw step input, the force bias in the F_x changes and a non-zero force appears for F_y. A similar exchange of forces is apparent following the second yaw step input. This force exchange is a consequence of unknown wind in the identified outdoor models coupled with a natural transfer of force from x to y and vice versa due to inertia following such a yawing manoeuvre with non-zero velocities. As it is difficult to visualize what these force changes entail, fig. 39 depicts the velocity response of the quadrotor, for which the wind direction dependencies of F_x and F_y are especially prominent. Moreover, the force response of F_z exhibits spikes in force that are in-line with the yaw step transitions. A positive force is a result of the aforementioned net increase in average rotor speeds induced by the controller to facilitate a yaw angle change. As such, this increase in force is reasonable and reflects expectations. Likewise, the moment responses are rational. Both M_x and M_y are predominately near-zero and show minimal responses. The minute exictations to M_x and M_y occur around regions of yaw step transitions, where the rotor speeds become saturated due to the altitude controller lowering the overall thrust. Spikes in the yawing moment, M_z, facilitate the yaw step transitions. These peaks are wider than the M_x and M_y counterparts and relate to the relatively slower yaw dynamics. Again, the simulated M_z responses resemble those found in the identification data as shown in fig. 42. Hence, the moment responses are sensible and reflect the underlying physics of the quadrotor.

While the outdoor MetalBeetle polynomial models appear to capture some of the underlying quadrotor dynamics, the contamination of unknown wind is apparent and produces some erroneous force responses for F_x and F_y, modulated by the yaw angle. As such, it is unsuitable as a general outdoor model. Nevertheless, these results show promise for useful outdoor models, especially if wind is properly accounted for and velocity estimation accurate.

2. *Hybrid model of the MetalBeetle*

A detailed analysis of the hovering simulation of the hybrid outdoor MetalBeetle model can be found in section VIII.B.2. Briefly, these hovering flight simulations show clear model instability whereby the ANN compensators appear to propagate the runaway velocities of the underlying polynomial force models to the rest of the models. Subsequently, the rotor speeds saturate while attempting to maintain control of the quadrotor and ultimately result in a loss of control. As aforementioned, the hovering flight is out of the domain of validity of the identified models. However, it nonetheless provides valuable insights on how the ANN compensators interact with the underlying polynomials and the resultant influence on the aggregate model predictions. A more representative evaluation of the hybrid model's performance is to simulate its response to oscillating attitude step inputs, as is done in the identification data.

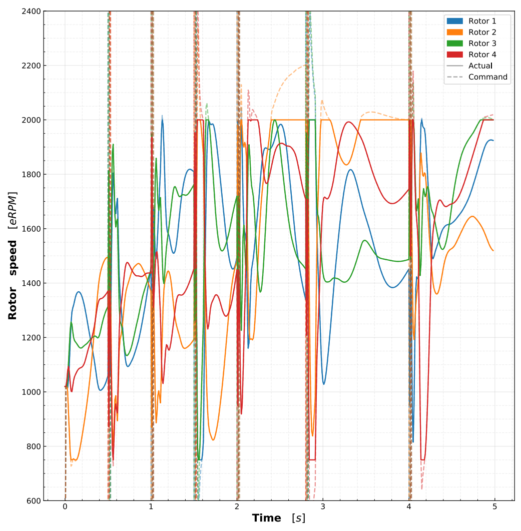


Fig. 47 Commanded rotor speed of the identified outdoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

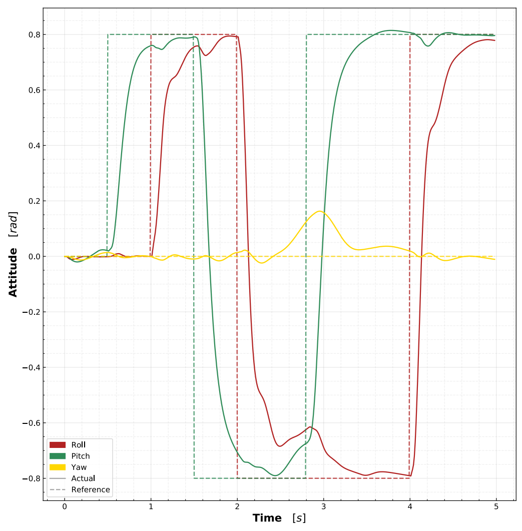


Fig. 48 Attitude response of the identified outdoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

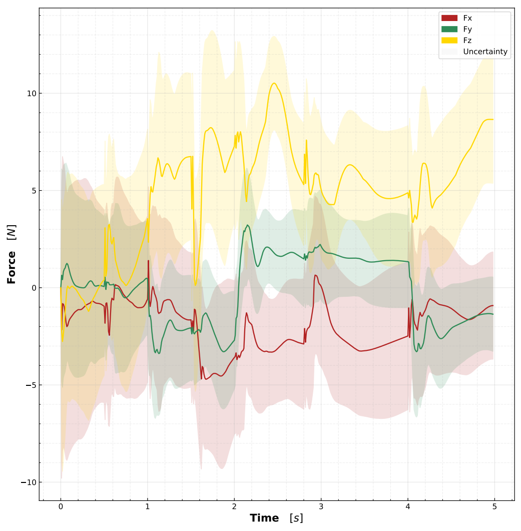


Fig. 49 Force response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

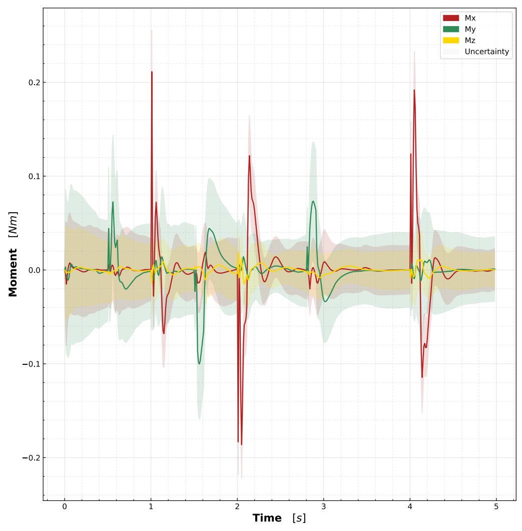


Fig. 50 Moment response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

Accordingly, the simulated response of the outdoor MetalBeetle hybrid model to oscillating roll and pitch step inputs is summarized in figs. 47 to 50. By virtue of these oscillating step inputs, the hybrid model is able to maintain stability. The attitude response, illustrated in fig. 48, shows that the hybrid model is able to satisfactorily track the demanding reference signal. Several kinks in the attitude response are observable and correspond to instances where the commanded rotor speeds encounter the saturation limits, either through a commanded step response itself or to maintain control. Consider, for example, the saturation of rotor 2 at around 2.8 seconds in fig. 47 which coincides with a loss of tracking for the roll and pitch angles and a induces yaw response. These responses parallel those found for the equivalent polynomial model. Aside from the apparent saturation of the commanded rotor speeds - which should not occur so

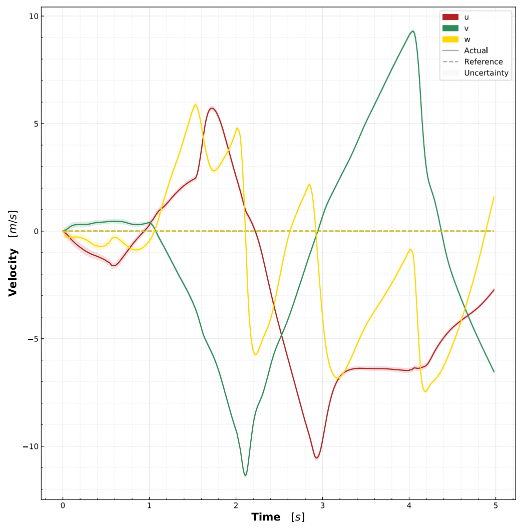


Fig. 51 Velocity response of the identified outdoor hybrid model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

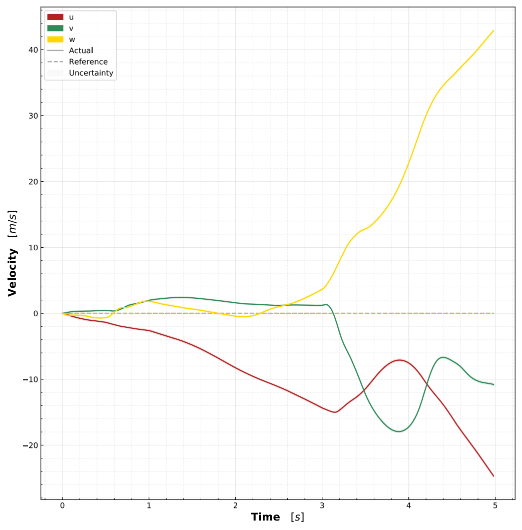


Fig. 52 Velocity response of the identified outdoor hybrid model of the MetalBeetle when subject to oscillating yaw angle step inputs.

easily in reality - the rotor speed responses appear sensible. Appropriate spikes in the commanded rotor speed occur to motivate attitude changes and the average rotor speeds increase for non-zero roll and pitch in order to maintain altitude.

The corresponding force responses are depicted in fig. 49. Again the predominately negative bias - likely induced by the unknown wind - of the underlying polynomial is apparent in the response of F_x. To visualize the effects of this force, the associated velocity response is shown in fig. 51. Indeed, the velocity is reminiscent of that of the underlying polynomial. Again, despite the predominately negative F_x, the positive velocity is afforded by the effects of gravity in the simulation as the quadrotor pitches. As with the underlying polynomial model, the simulated F_x response does not resemble the measured F_x for similar manoeuvres (refer to fig. 40) and highlights the detrimental effects of wind for the models. Furthermore, in comparison to the underlying polynomial, similarities between the measured F_y and the simulated response fade for the hybrid model. While the hybrid model undoubtedly draws inspiration from the measured data in fig. 41, it affords a more step-like signal which is less reflective of the true measurements than the underlying polynomial. However, this may be due to the greater rotor saturation experienced by the hybrid model. Given the poor velocity estimations of the GPS module, the simulated velocity response along y, v, does not match the measurement data. However, it is nonetheless similar to the velocities seen for the indoor measurement data (see fig. 129 in section VIII.D), which lends some support to the validity of the hybrid F_y model. Similarly, the hybrid F_z model sensibly demonstrates an increase in force for the duration of the step inputs to maintain altitude.

Likewise, the simulated moment responses to the oscillating roll and pitch step inputs - illustrated in fig. 50 - are realistic. Indeed, the rapid sine-wave like moment excitations are observable in the simulated responses of M_x and M_y while the M_z moment remains largely near-zero. Note that the difference in magnitude between the responses of M_x and M_y peaks is, in part, due to differences in the moment of inertia. However, the contamination by wind may also contribute to this discrepancy. The simulated moments, including this magnitude difference, are in accordance with the measured moment responses for similar inputs observable in outdoor flights, as shown in fig. 42. These responses are also in agreement with the polynomial model simulation.

In contrast, the simulated response of the hybrid model to oscillating yaw step inputs does not resemble the underlying polynomial. Indeed, the attitude response of the quadrotor during this manoeuvre, illustrated in fig. 54, demonstrates that it is able to track the first yaw step but fails to do so with the second step. Interestingly, the tracking of the first yaw input is better than the underlying polynomial but that of the second is much worse. Moreover, during the second yaw step command, the other attitudes diverge from zero, which is undesirable. The reasons behind this become apparent in the commanded rotor speed responses, depicted in fig. 53, which show the saturation of multiple rotors just before and during the second step input.

Again, the ANN component of the hybrid models likely instigates the divergence of the attitudes through the

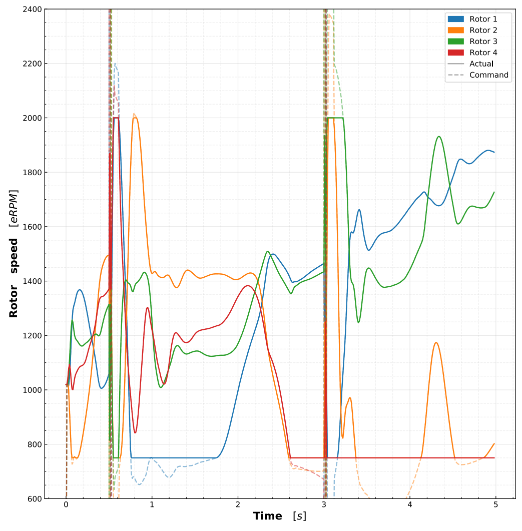


Fig. 53 Commanded rotor speed of the identified outdoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

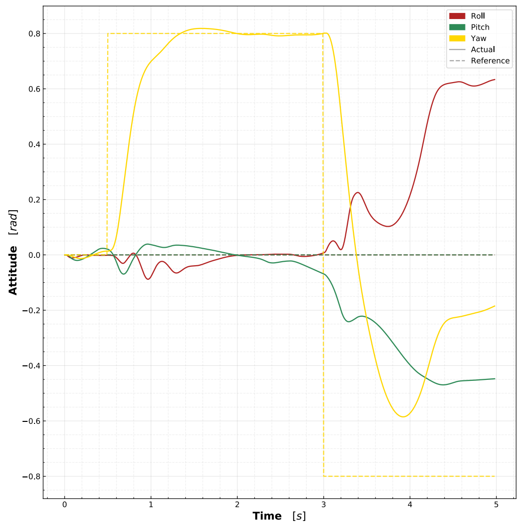


Fig. 54 Attitude response of the identified outdoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

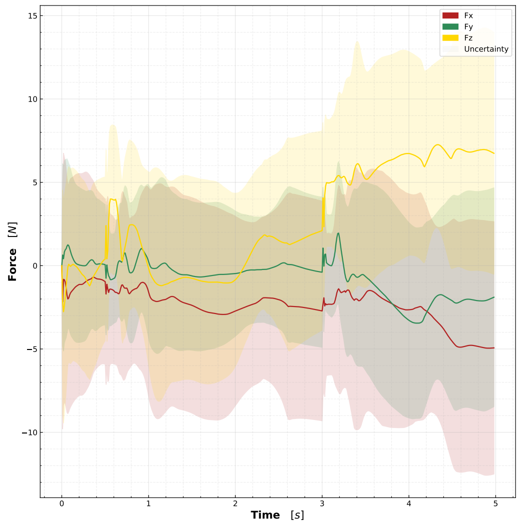


Fig. 55 Force response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

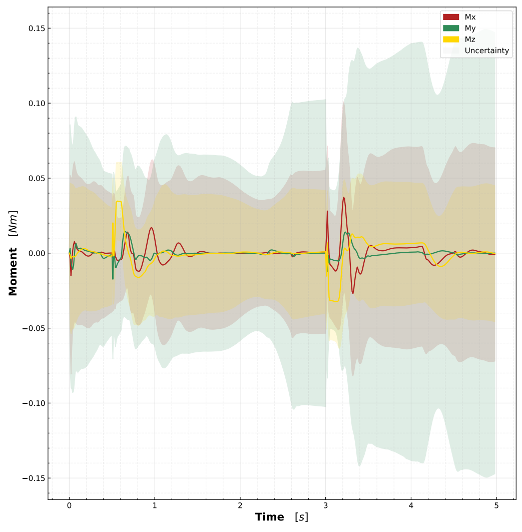


Fig. 56 Moment response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

propagation of runaway velocity terms to the moment models, culminating in erroneous predictions. Indeed, through the simulated force response for oscillating yaw step inputs in fig. 55, the negative bias of F_x is obvious and induces a runaway u velocity as shown in fig. 52. The F_y model mostly behaves until the onset of the instabilities. However, unlike the underlying polynomial model, the obvious exchanges in force between F_x and F_y due the first yaw manoeuvre are absent. This is also apparent in the velocity responses in fig. 52. This may be due to a lack of response in the roll and pitch attitudes during the yaw manoeuvre for the hybrid simulation. However, due to a significant change in yaw for the second input, an exchange of velocities between u and v is observed in fig. 52. Although, this manoeuvre appears to instigate instability in the model, accentuating the issues of the hybrid model. These results highlight the significant

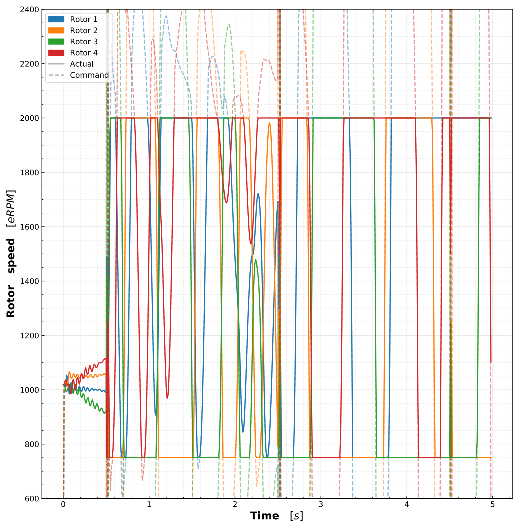


Fig. 57 Commanded rotor speed of the identified outdoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

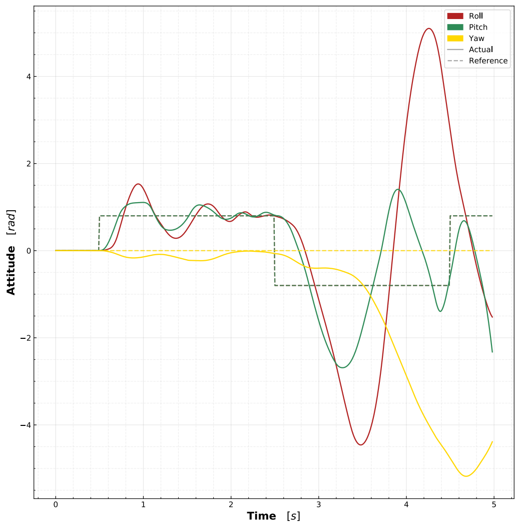


Fig. 58 Attitude response of the identified outdoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

influence of the ANN compensator on the hybrid model predictions. As with the other forces, the F_z become unstable following the second yaw step command and stimulates a runaway velocity, w, as seen in fig. 52. However, F_z exhibits an expected increase in force during the first yaw step input due to the net increase in rotor speeds induced by the yawing manoeuvre.

Likewise, prior to the instabilities, the moment responses depicted in fig. 56 reflect expectations. M_x and M_y predominantly remain near-zero while M_z exhibits the characteristic sine-like pulse of a moment response. Note that the simulated M_z peak is wider than the measurement data, in fig. 42. This is likely due to the relatively long rise time of the simulated yaw response, especially in comparison to the faster roll and pitch responses. This can be ironed out with a better controller, through an increased proportional P gain. Interestingly, the associated PIs of the moment models appear to widen in anticipation the instabilities, emphasizing the utility of these PIs.

Overall, the simulation results of the identified outdoor MetalBeetle hybrid model reveal obvious instabilities and negate its validity and utility. While the biased F_x of the underlying polynomial model is also invalid, the ANN components of the hybrid model propagate these instabilities throughout the rest of the hybrid model, rendering it worse than its underlying polynomial. As such, it may be valuable to investigate strategies to mitigate this effect in the ANN models, such that malicious states do not contaminate all the models.

3. ANN model of the MetalBeetle

The hovering simulations of the identified ANN model of the outdoor MetalBeetle are outlined in depth in section VIII.B.3. Through these simulations, it is found that the F_z ANN model produces an unrealistic thrust profile (refer to fig. 88) wherein the lowest predicted forces are much greater than zero. Subsequently, the quadrotor is constantly accelerating upwards, even with non-zero attitudes[‡‡‡], and quickly leads to runaway velocities. As with the hybrid model, due to the dense architecture of the ANN models, these runaway terms dominate the model predictions and induce instability. Although the input states for the ANN models are normalized, the erroneous thrust model itself may arise from asymmetries in the F_z identification data, which has mostly positive thrust.

While this may be argued as a byproduct of evaluating the ANN model's response beyond its domain of validity, the unstable characteristics of the outdoor MetalBeetle ANN model persist in the simulations of oscillating attitude step inputs even though such inputs are more representative of the identification data. Therefore, the identified outdoor MetalBeetle ANN model is likely invalid altogether.

The futile efforts of the PID controller to maintain control of the quadrotor are apparent in the incredibly oscillatory

[‡‡‡]Confined to a region such that the quadrotor is always upright (i.e. not upside-down).

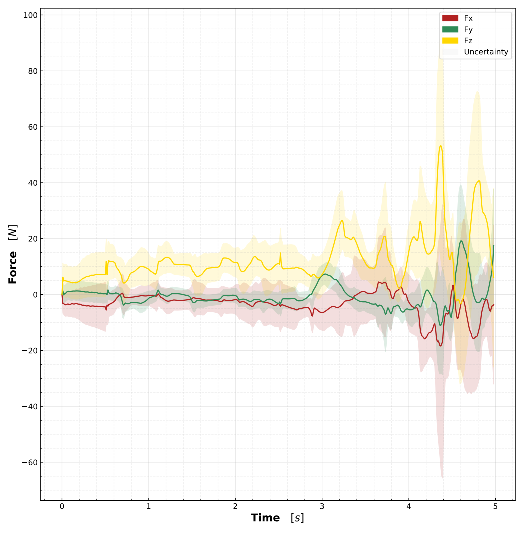


Fig. 59 Force response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

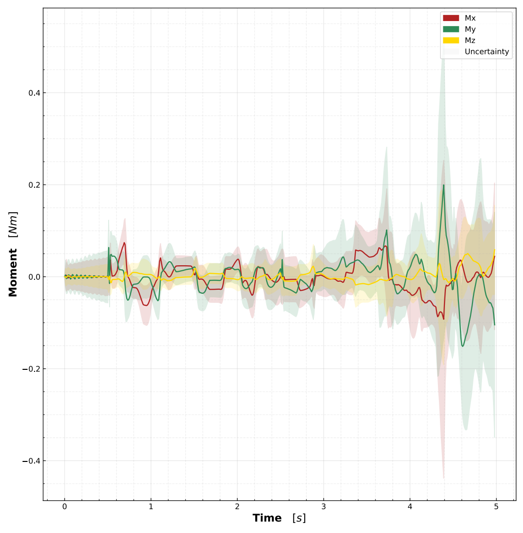


Fig. 60 Moment response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

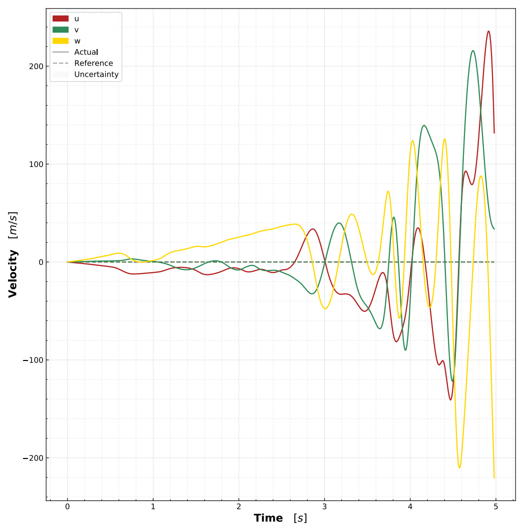


Fig. 61 Velocity response of the identified outdoor ANN model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

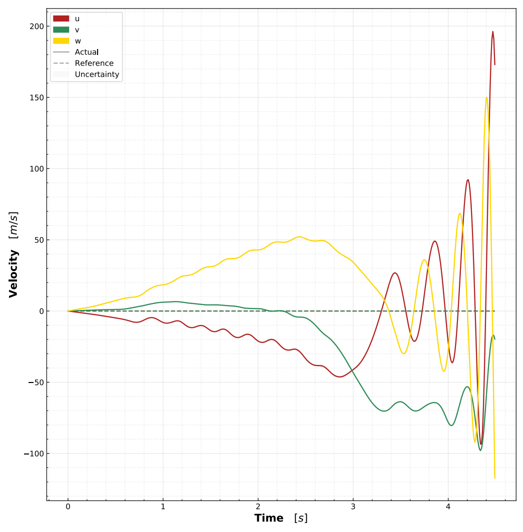


Fig. 62 Velocity response of the identified outdoor ANN model of the MetalBeetle when subject to oscillating yaw angle step inputs.

nature of the commanded rotor speeds, depicted in fig. 57 oscillating roll & pitch step inputs and fig. 63 for oscillating yaw step inputs. The commanded rotor speeds spend most of their time in a saturated state and primarily oscillate between the saturation bounds. Such behaviour is undesirable and in disagreement with the adequate control of the polynomial models. Nevertheless, the outdoor MetalBeetle ANN model is initially able to follow the commanded step inputs, as shown in fig. 58 for oscillating roll & pitch step inputs and fig. 64 for oscillating yaw step inputs, albeit with poor damping characteristics for the roll and pitch angles[§§§]. However, after this initial success, the ANN model is

[§§§]Note that a trade-off between high frequency rotor speed changes and low frequency tracking oscillations is made as both are influenced by the derivative component of the PID controller.

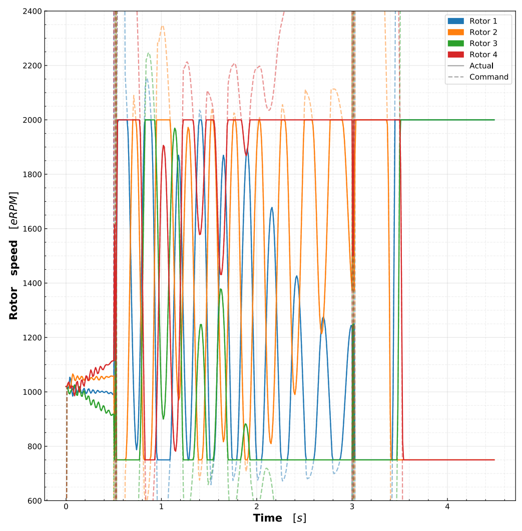


Fig. 63 Commanded rotor speed of the identified outdoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

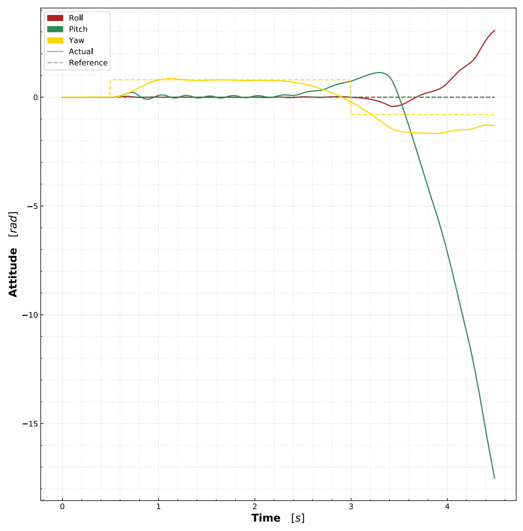


Fig. 64 Attitude response of the identified outdoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

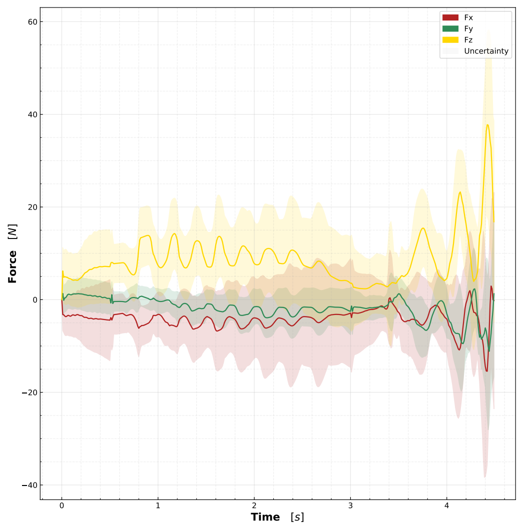


Fig. 65 Force response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

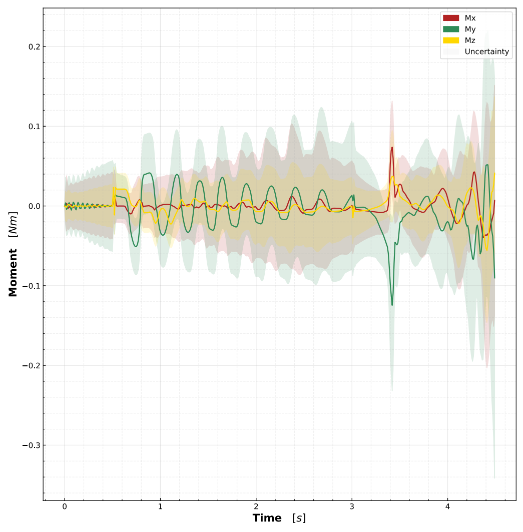


Fig. 66 Moment response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

unable to track the subsequent attitude step commands and instead overshoots considerably. While the performance can be improved with a better PID controller, the rotor speeds are already much too saturated and imply that the ANN model is difficult to control.

This is evident in the force responses of the oscillating attitude step inputs depicted in fig. 59 for roll & pitch inputs and fig. 65 for yaw inputs. In both cases, the force responses of F_x and F_y appear to respond appropriately to the attitude step inputs by exhibiting corresponding oscillations. Moreover, in accordance with the polynomial and hybrid equivalents, the F_z responses also increase following attitude changes. However, the persistent biases in the force models, particularly for F_z, induce runaway velocities (see fig. 61 and fig. 62) and initiate model instability. For example, with the force responses to oscillating roll & pitch inputs (fig. 59), the F_x rarely rises above zero newtons and the F_z model exhibits a positive force consistently above 5 N. For reference, the weight of the quadrotor is 0.396 kg ($\approx$ 4 N). Hence, the predicted F_z force is much too high. The subsequent runaway velocities quickly render the ANN models' predictions invalid as they extend beyond the domain of validity of the model and propagate instability through all the densely constructed ANN models. The rapid changes in velocity are also quite unrealistic given the apparent magnitudes - around 50 to 100 N - at play.

Likewise, the moment model responses of the oscillating roll & pitch step inputs, shown in fig. 60, and yaw step inputs, given by fig. 66, reveal the oscillatory roots of the unstable attitudes. These responses also reflect the instabilities ubiquitous throughout the model. Again, the associated PIs appear to widen with the growing model instabilities, accentuating the poor confidence in model predictions. This widening of PIs is particularly prominent for the oscillating yaw step inputs.

Contrary to the relatively good performance metrics enjoyed by the ANN model of the outdoor MetalBeetle, the simulation results imply that this model harbors little utility and demonstrates unrealistic predictions. Again, perhaps not all of the constituent ANN models are poor, but the F_z ANN model is definitely invalid. It is suspected that most, if not all, of the outdoor MetalBeetle ANN models' infeasible responses propagate from this offending F_z model.

VII. Conclusion

In this paper, outdoor quadrotor models are identified which accurately capture high-speed flight of up to 19 ms^{-1} and aggressive manoeuvres - such as punch-outs, flips, and barrel rolls - despite the contamination of unknown wind for the first time. This is facilitated through the development of a modular system identification pipeline that is compatible with outdoor high-speed flight and aggressive manoeuvring along with a complementary simulation platform employed to validate the identified models. Drawing inspiration from previous successes in quadrotor model identification, polynomial stepwise regression, artificial neural networks (ANNs), and a novel hybrid approach first proposed in this paper combining the two techniques are implemented in the pipeline to identify the outdoor models of the MetalBeetle. Moreover, the subsequent model predictions are accompanied by prediction intervals (PIs) which describe the models' confidence in their predictions. From the simulation results, the estimated PIs are found to reflect growing uncertainty in the model inputs and may forecast model instability.

One of the challenges faced with outdoor model identification is the accurate estimation of the quadrotor's state, especially the linear velocity. Here, a commercially available GPS module is employed for linear velocity and position estimation of the MetalBeetle. Unfortunately, strong wind of up to 6 ms^{-1} with gusts of up to 10 ms^{-1} plagued outdoor data acquisition. Through an analysis of the model structures and sensitivities, the contamination of wind is evident for the outdoor models through an explicit dependence on the yaw angle. Nonetheless, the outdoor models are still capable of producing faithful representations of the measured aerodynamic forces and moments, particularly during aggressive manoeuvring such as punch-outs, barrel rolls, front-flips and back-flips. In fact, much of the modelling success enjoyed by the outdoor models is unrelated to the velocity and instead attributable to emergent effects associated with the total rotor speed. Therefore, this term is likely essential for explaining the underlying dynamics.

Simulations of the identified models' responses to oscillating attitude step commands show mixed results between the system identification techniques. The polynomial and hybrid models' responses to oscillating roll and pitch step inputs are reminiscent of the measurement data when subject to similar inputs, hence some fundamental outdoor dynamics have been recognized by these models despite the wind contamination. Nonetheless, the effects of wind are still apparent in the step attitude responses through non-zero force biases. The hybrid model fails to track yaw step inputs whereas the polynomial does not. Therefore, it is found that the hybrid approach, as configured in this paper, does not achieve compelling performance improvements over the underlying polynomial model. In fact, the corresponding hybrid models only propagate the wind contaminated polynomial force model errors throughout the rest of the model through the dense ANN architecture, culminating in instabilities due to quickly saturating rotor speeds. Likewise, the ANN-only outdoor models suffer from an unrealistic F_z model that produces erroneous thrust predictions, which also subsequently permeate instability through the rest of the ANN models. Consequently, the identified ANN models fail to track any of the commanded attitude step inputs and are found to be completely invalid, despite seemingly good model fits. Therefore, contrary to expectations, the outdoor polynomial model produces the most realistic responses as it is able to appropriately track oscillating roll, pitch, and yaw step inputs and exhibited similar force, velocity, and moment responses as the measurement data subject to similar input commands. Furthermore, through these simulations, the PIs for all models were found to increase with the growing uncertainty in the system and appear to forecast instability. This emphasizes the utility of the PIs. Given the results of the simulation and obvious contamination of wind, the identified models cannot be considered valid or useful for general simulation. Nevertheless, the polynomial models are the most practical for modelling as they are easy to interpret, not computationally demanding, and intuitive to tune controllers for. Therefore, these results promote the use of the polynomial models for outdoor quadrotor identification.

While the results of this paper show promise for the developed system identification pipeline, they also give rise to numerous recommendations and future avenues of research. Perhaps the most pressing concern is the development of better velocity estimation schemes for outdoor applications. To this end, future research should investigate and validate airspeed sensors, such as pitot tubes, to facilitate velocity estimation as the GPS sensors are evidently insufficient. Until then, future outdoor flights should be conducted in windless conditions for reliable data. Related to data collection, more manoeuvres which excite the moments, particularly the yawing moment, should be identified and conducted given their sparsity in the current data sets. Indeed, all employed system identification techniques struggled with the moment model identification. Moreover, many interesting effects consistently emerge in the polynomial model structures for the outdoor models. Hence, the significance of these effects, and perhaps their relation to physical phenomena, should be examined in future work. This may lead to the discovery of potentially new, quadrotor-specific, fundamental modelling terms. Accordingly, further avenues of research pertaining to improving the models would be to develop rotor-local models which may better capture interactions effects. Furthermore, reasons explaining the apparent difficulty of the ANN models to identify an appropriate F_z model should be investigated. Given that the hybrid approach suffers from an already invalid underlying polynomial model, its performance with respect to a valid polynomial model should be determined to truly establish the feasibility of this approach. Indeed, alternative hybrid approaches may also be prototyped, such as the combination of model constituents from different system identification techniques. For example, the use of the ANN F_x and polynomial F_z in tandem. Similarly, more advanced system identification techniques, such as simplex splines, may be beneficial to address issues arising from different flight domains. Given the simplicity of the ANN used here, alternative ANN configurations, for example recurrent neural networks, and structures may also be implemented in the pipeline. Finally, in order to make further advancements towards a functional online system identification algorithm, the computational complexities and feasibility of the system identification techniques for real-time model identification should be determined, including the subsequent resource and hardware requirements these techniques impose. Nonetheless, the results of this paper make confident strides towards the advancement of quadrotor models.

References

[1] Verbeke, J., and Schutter, J. D., "Experimental maneuverability and agility quantification for rotary unmanned aerial vehicle," *International Journal of Micro Air Vehicles*, Vol. 10, No. 1, 2018, pp. 3–11. https://doi.org/https://doi.org/10.1177/1756829317736204, URL https://journals.sagepub.com/doi/full/10.1177/1756829317736204.

[2] Hoffmann, G., Huang, H., Waslander, S., and Tomlin, C., "Quadrotor helicopter flight dynamics and control: Theory and experiment," *AIAA guidance, navigation and control conference and exhibit*, 2007, p. 6461. https://doi.org/https://doi.org/10.2514/6.2007-6461.

[3] Mellinger, D., Michael, N., and Kumar, V., "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, Vol. 31, No. 5, 2012, pp. 664–674. https://doi.org/10.1177/0278364911434236, URL https://doi.org/10.1177/0278364911434236.

[4] Molenkamp, D., Kampen, E.-J. V., de Visser, C. C., and Chu, Q. P., *Intelligent Controller Selection for Aggressive Quadrotor Manoeuvring*, 2017. https://doi.org/10.2514/6.2017-1068, URL https://arc.aiaa.org/doi/abs/10.2514/6.2017-1068.

[5] Loianno, G., Brunner, C., McGrath, G., and Kumar, V., "Estimation, Control, and Planning for Aggressive Flight with a Small Quadrotor with a Single Camera and IMU," *IEEE Robotics and Automation Letters*, Vol. 2, 2017, pp. 404–411. https://doi.org/10.1109/LRA.2016.2633290.

[6] Kaufmann, E., Loquercio, A., Ranftl, R., Müller, M., Koltun, V., and Scaramuzza, D., "Deep drone acrobatics," *arXiv preprint arXiv:2006.05768*, 2020.

[7] Sun, S., Schilder, R. J., and de Visser, C. C., *Identification of Quadrotor Aerodynamic Model from High Speed Flight Data*, 2018. https://doi.org/10.2514/6.2018-0523, URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0523.

[8] Sun, S., de Visser, C., and Chu, Q., "Quadrotor Gray-Box Model Identification from High-Speed Flight Data," *Journal of Aircraft*, Vol. 56, No. 2, 2019, pp. 645–661. https://doi.org/10.2514/1.C035135, URL https://doi.org/10.2514/1.C035135.

[9] Li, S., Wagter, C. D., de Visser, C., Chu, Q., and de Croon, G., "In-flight model parameter and state estimation using gradient descent for high-speed flight," *International Journal of Micro Air Vehicles*, Vol. 11, 2019, p. 1756829319833685. https://doi.org/10.1177/1756829319833685, URL https://doi.org/10.1177/1756829319833685.

[10] Sun, S., and de Visser, C., "Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime," *IEEE Robotics and Automation Letters*, Vol. 4, No. 4, 2019, pp. 3868–3875. https://doi.org/10.1109/LRA.2019.2928758.

[11] Bauersfeld, L., Kaufmann, E., Foehn, P., Sun, S., and Scaramuzza, D., "NeuroBEM: Hybrid Aerodynamic Quadrotor Model," *CoRR*, Vol. abs/2106.08015, 2021. URL https://arxiv.org/abs/2106.08015.

[12] Sun, S., Sijbers, L., Wang, X., and de Visser, C., "High-Speed Flight of Quadrotor Despite Loss of Single Rotor," *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, 2018, pp. 3201–3207. https://doi.org/10.1109/LRA.2018.2851028.

[13] Smeur, E. J. J., Chu, Q., and de Croon, G. C. H. E., "Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Air Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 3, 2016, pp. 450–461. https://doi.org/10.2514/1.G001490, URL https://doi.org/10.2514/1.G001490.

[14] Bansal, S., Akametalu, A. K., Jiang, F. J., Laine, F., and Tomlin, C. J., "Learning quadrotor dynamics using neural network for flight control," Institute of Electrical and Electronics Engineers Inc., 2016, pp. 4653–4660. https://doi.org/10.1109/CDC.2016.7798978.

[15] Bagherzadeh, S. A., "Nonlinear aircraft system identification using artificial neural networks enhanced by empirical mode decomposition," *Aerospace Science and Technology*, Vol. 75, 2018, pp. 155–171. https://doi.org/10.1016/j.ast.2018.01.004.

[16] Mohajerin, N., Mozifian, M., and Waslander, S., "Deep Learning a Quadrotor Dynamic Model for Multi-Step Prediction," Institute of Electrical and Electronics Engineers Inc., 2018, pp. 2454–2459. https://doi.org/10.1109/ICRA.2018.8460840.

[17] Lu, S., and Başar, T., "Robust nonlinear system identification using neural-network models," *IEEE Transactions on Neural Networks*, Vol. 9, 1998, pp. 407–429. https://doi.org/10.1109/72.668883.

[18] Chen, S., Billings, S. A., and Grant, P. M., "Non-linear system identification using neural networks," *International Journal of Control*, Vol. 51, 1990, pp. 1191–1214. https://doi.org/10.1080/00207179008934126, URL https://www.tandfonline.com/doi/abs/10.1080/00207179008934126.

[19] Gevrey, M., Dimopoulos, I., and Lek, S., "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecological Modelling*, Vol. 160, No. 3, 2003, pp. 249–264. https://doi.org/https://doi.org/10.1016/S0304-3800(02)00257-0, URL https://www.sciencedirect.com/science/article/pii/S0304380002002570.

[20] Scardi, M., and Harding, L. W., "Developing an empirical model of phytoplankton primary production: a neural network case study," *Ecological Modelling*, Vol. 120, No. 2, 1999, pp. 213–223. https://doi.org/https://doi.org/10.1016/S0304-3800(99)00103-9, URL https://www.sciencedirect.com/science/article/pii/S0304380099001039.

[21] Sun, S., Wang, X., Chu, Q., and d. Visser, C., "Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors," *IEEE Transactions on Robotics*, 2020, pp. 1–15. https://doi.org/10.1109/TRO.2020.3010626.

[22] Pearce, T., Zaki, M., Brintrup, A., and Neely, A., "High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach," , 2018. URL https://arxiv.org/abs/1802.07167.

[23] Sun, S., Baert, M., van Schijndel, B. S., and de Visser, C. C., "Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4273–4279. https://doi.org/10.1109/ICRA40945.2020.9197239.

[24] Svacha, J., Loianno, G., and Kumar, V., "Inertial Yaw-Independent Velocity and Attitude Estimation for High-Speed Quadrotor Flight," *IEEE Robotics and Automation Letters*, Vol. 4, No. 2, 2019, pp. 1109–1116. https://doi.org/10.1109/LRA.2019.2894220.

[25] Bouabdallah, S., Murrieri, P., and Siegwart, R., "Design and Control of an Indoor Micro Quadrotor," *Proc. of The International Conference on Robotics and Automation (ICRA)*, Vol. 5, ETH-Zürich, [2004], Zürich, 2004, pp. 4393 – 4398. https://doi.org/10.3929/ethz-a-010085499, iEEE International Conference on Robotics and Automation (ICRA 2004); Conference Location: New Orleans, LA, USA; Conference Date: 26 April-1 May, 2004.

[26] Svacha, J., Mohta, K., and Kumar, V., "Improving quadrotor trajectory tracking by compensating for aerodynamic effects," *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017, pp. 860–866. https://doi.org/10.1109/ICUAS.2017.7991501.

[27] Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., and Kumar, V., “Influence of aerodynamics and proximity effects in quadrotor flight,” *Experimental robotics*, Springer, 2013, pp. 289–302.

[28] Huang, H., Hoffmann, G. M., Waslander, S. L., and Tomlin, C. J., “Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering,” *2009 IEEE international conference on robotics and automation*, IEEE, 2009, pp. 3277–3282. https://doi.org/10.1109/ROBOT.2009.5152561.

[29] Johnson, W., *Helicopter Theory*, Dover Books on Aeronautical Engineering, Dover Publications, 2012. URL https://books.google.nl/books?id=FiEapaNgjLcC.

[30] Abeywardena, D. M. W., Kodagoda, S., Dissanayake, G., and Munasinghe, S. R., “Improved State Estimation in Quadrotor MAVs: A Novel Drift-Free Velocity Estimator,” *CoRR*, Vol. abs/1509.03388, 2015. URL http://arxiv.org/abs/1509.03388.

[31] Mahony, R., Kumar, V., and Corke, P., “Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor,” *IEEE Robotics Automation Magazine*, Vol. 19, No. 3, 2012, pp. 20–32. https://doi.org/10.1109/MRA.2012.2206474.

[32] Klein, V., and Morelli, E. A., *Aircraft System Identification: Theory and Practice*, 2006. https://doi.org/10.2514/4.861505.

[33] Morelli, E. A., “Global nonlinear aerodynamic modeling using multivariate orthogonal functions,” *Journal of Aircraft*, Vol. 32, No. 2, 1995, pp. 270–277. https://doi.org/10.2514/3.46712, URL https://doi.org/10.2514/3.46712.

[34] Zeiler, M. D., and Fergus, R., “Visualizing and understanding convolutional networks,” *European conference on computer vision*, Springer, 2014, pp. 818–833.

[35] Hinton, G. E., and Salakhutdinov, R. R., “Reducing the dimensionality of data with neural networks,” *Science*, Vol. 313, 2006, pp. 504–507. https://doi.org/10.1126/science.1127647, URL http://arxiv.org/abs/physics/0601055www.sciencemag.org/cgi/content/full/313/5786/502/DC1.

[36] Kirkpatrick, K., May, J., and Valasek, J., “Aircraft system identification using Artificial Neural Networks,” American Institute of Aeronautics and Astronautics Inc., 2013. https://doi.org/10.2514/6.2013-878.

[37] Harris, J., Arthurs, F., Henrickson, J. V., and Valasek, J., “Aircraft system identification using artificial neural networks with flight test data,” Institute of Electrical and Electronics Engineers Inc., 2016, pp. 679–688. https://doi.org/10.1109/ICUAS.2016.7502624.

[38] Kumar, R., Ganguli, R., and Omkar, S. N., “Rotorcraft parameter estimation using radial basis function neural network,” *Applied Mathematics and Computation*, Vol. 216, 2010, pp. 584–597. https://doi.org/10.1016/j.amc.2010.01.081.

[39] Punjani, A., and Abbeel, P., “Deep learning helicopter dynamics models,” Institute of Electrical and Electronics Engineers Inc., 2015, pp. 3223–3230. https://doi.org/10.1109/ICRA.2015.7139643.

[40] Bakshi, N. A., and Ramachandran, R., “Indirect model reference adaptive control of quadrotor UAVs using neural networks,” Institute of Electrical and Electronics Engineers Inc., 2016. https://doi.org/10.1109/ISCO.2016.7727123.

[41] Al-Mahasneh, A. J., Anavatu, S. G., and Garratt, M., “Nonlinear multi-input multi-output system identification using neuro-evolutionary methods for a quadcopter,” Institute of Electrical and Electronics Engineers Inc., 2017, pp. 217–222. https://doi.org/10.1109/ICACI.2017.7974512.

[42] Heaton, J., *Introduction to Neural Networks with Java*, Heaton Research, 2008. URL https://books.google.nl/books?id=Swlcw7M4uD8C.

[43] Werbos, P. J., “Backpropagation through time: what it does and how to do it,” *Proceedings of the IEEE*, Vol. 78, No. 10, 1990, pp. 1550–1560. https://doi.org/10.1109/5.58337.

[44] Krizhevsky, A., Sutskever, I., and Hinton, G. E., “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, Vol. 60, 2017, pp. 84–90. https://doi.org/10.1145/3065386.

[45] Lecun, Y., Bengio, Y., and Hinton, G., “Deep learning,” *Nature*, Vol. 521, 2015, pp. 436–444. https://doi.org/10.1038/nature14539.

[46] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., “Learning representations by back-propagating errors,” *Nature*, Vol. 323, 1986. https://doi.org/10.1038/323533a0.

[47] Kingma, D. P., and Ba, J., “Adam: A Method for Stochastic Optimization,” , 2017. URL https://arxiv.org/abs/1412.6980.

[48] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016. http://www.deeplearningbook.org.

[49] Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F., "Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances," *IEEE Transactions on Neural Networks*, Vol. 22, No. 9, 2011, pp. 1341–1356. https://doi.org/10.1109/TNN.2011.2162110.

[50] Armanini, S. F., Karásek, M., de Croon, G. C. H. E., and de Visser, C. C., "Onboard/Offboard Sensor Fusion for High-Fidelity Flapping-Wing Robot Flight Data," *Journal of Guidance, Control, and Dynamics*, Vol. 40, 2017, pp. 2121–2132. https://doi.org/10.2514/1.G002527, URL https://arc.aiaa.org/doi/10.2514/1.G002527.

VIII. Appendix

A. Outdoor MetalBeetle results: Extra

In this section, supplementary plots and tables associated with the results of the identified outdoor models of the MetalBeetle are presented.

1. Plots of error residuals of identified outdoor models

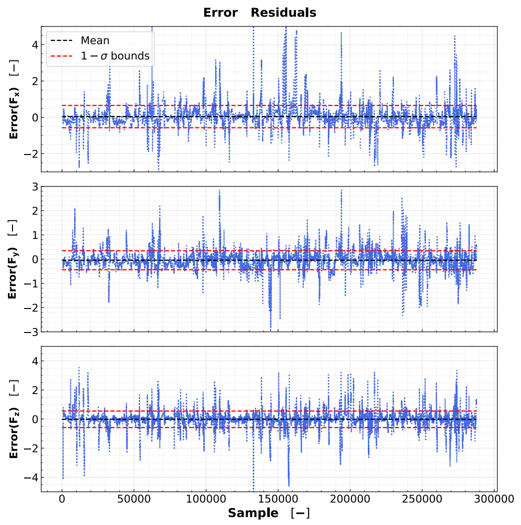


Fig. 67 Error residuals of the identified polynomial force models of the *outdoor* MetalBeetle.

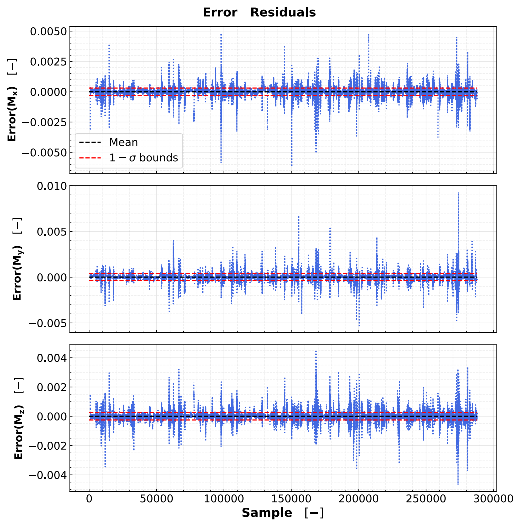


Fig. 68 Error residuals of the identified polynomial moment models of the *outdoor* MetalBeetle.

2. Plots of autocorrelation of error residuals of identified outdoor models

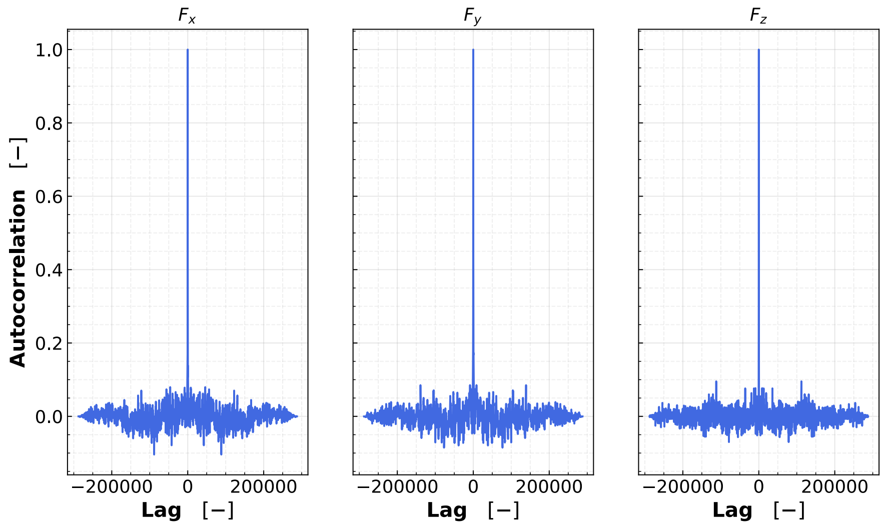


Fig. 69 Autocorrelation of the error residuals of the identified polynomial force models of the *outdoor* MetalBeetle.

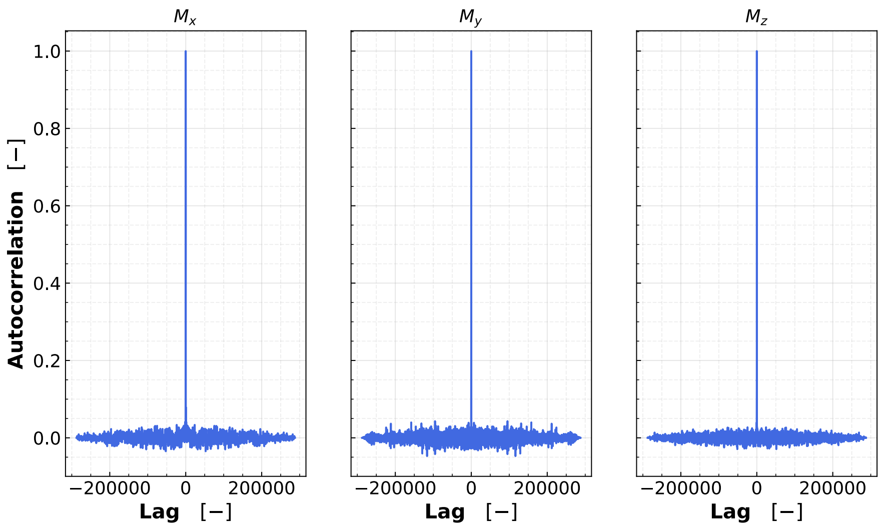


Fig. 70 Autocorrelation of the error residuals of the identified polynomial moment models of the *outdoor* MetalBeetle.

3. Full identified moment model performances

Table 4 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid moment models for the MetalBeetle (Outdoor flight). The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Mx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0300	0.0320	-0.7392	-0.7682	94.6799	93.9284	6.8186	7.4659
Poly	0.0258	0.0276	-0.2833	-0.3080	96.6600	95.9430	11.4636	11.3289
Hybrid	0.0252	0.0267	-0.2221	-0.2258	98.8612	98.5824	16.7412	16.9927

My								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0304	0.0321	0.0693	0.1323	95.1931	94.4377	8.0847	8.4023
Poly	0.0309	0.0335	0.0378	0.0556	97.8895	97.4668	21.8770	21.8470
Hybrid	0.0397	0.0415	-0.5905	-0.4516	99.0548	98.8535	26.3385	26.2501

Mz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0278	0.0278	0.0103	-0.0762	94.4592	94.3738	7.6167	8.5046
Poly	0.0355	0.0375	-0.6098	-0.9564	97.3349	97.0472	18.2845	19.7312
Hybrid	0.0346	0.0362	-0.5325	-0.8305	98.3357	98.1689	19.9518	21.6178

4. Plots of model predictions for validation flights

In this subsection, complementary plots of the force and moment predictions of unseen (i.e. validation) flights for MetalBeetle are shown. In particular, the remaining force and moment plots of the validation flight which are not illustrated in section V.

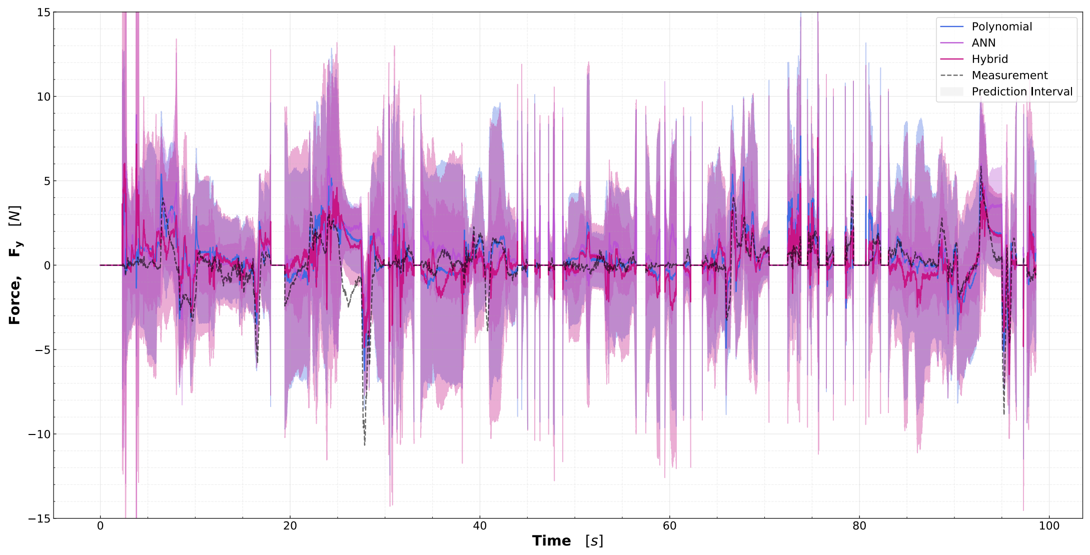


Fig. 71 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_y for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured F_y response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\,ms^{-1}$ with gusts of up to $\approx 10\,ms^{-1}$). The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind.

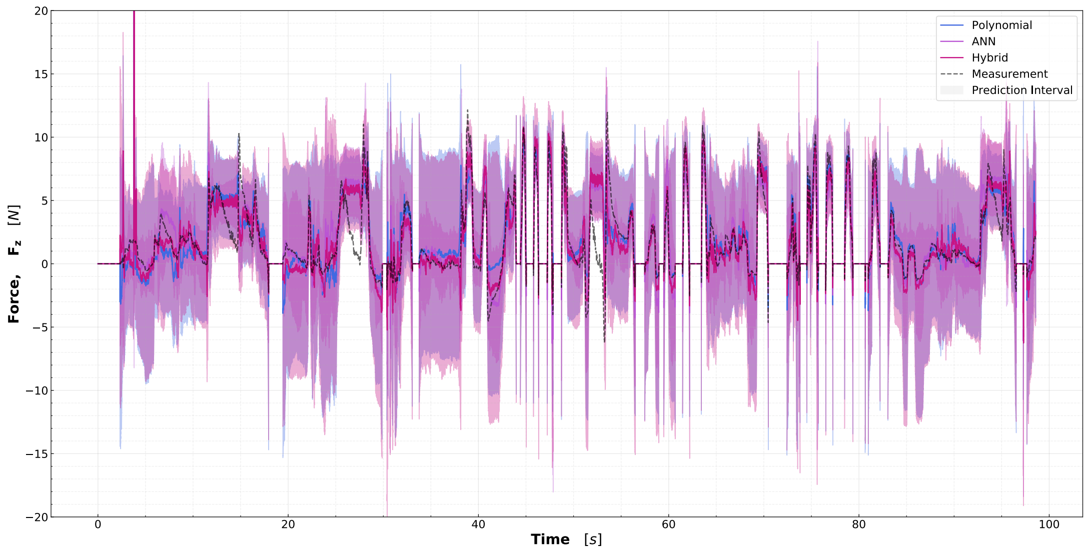


Fig. 72 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_z for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured F_z response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\, ms^{-1}$ with gusts of up to $\approx 10\, ms^{-1}$). The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind.

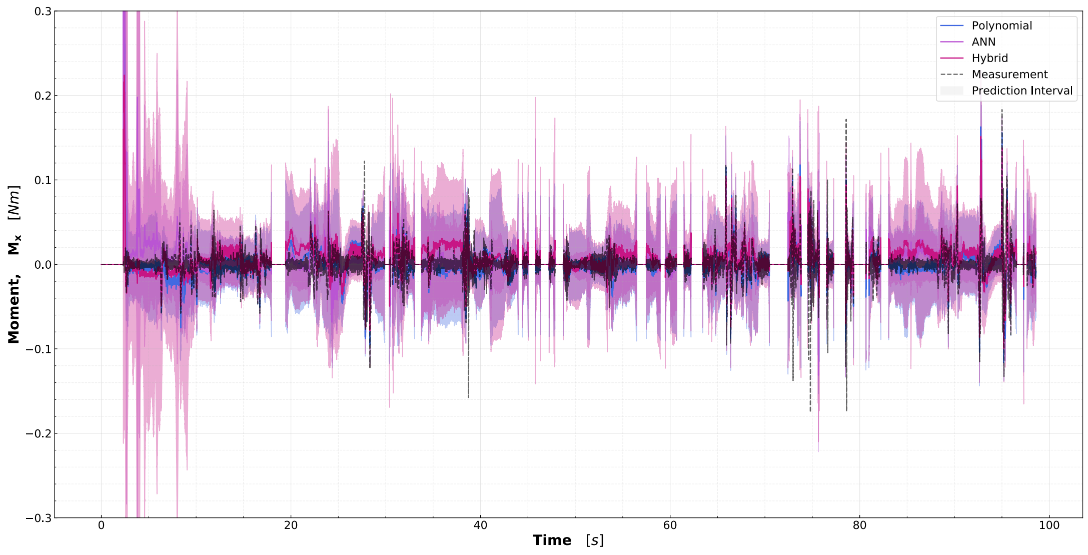


Fig. 73 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_x for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured M_x response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\, ms^{-1}$ with gusts of up to $\approx 10\, ms^{-1}$). The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind.

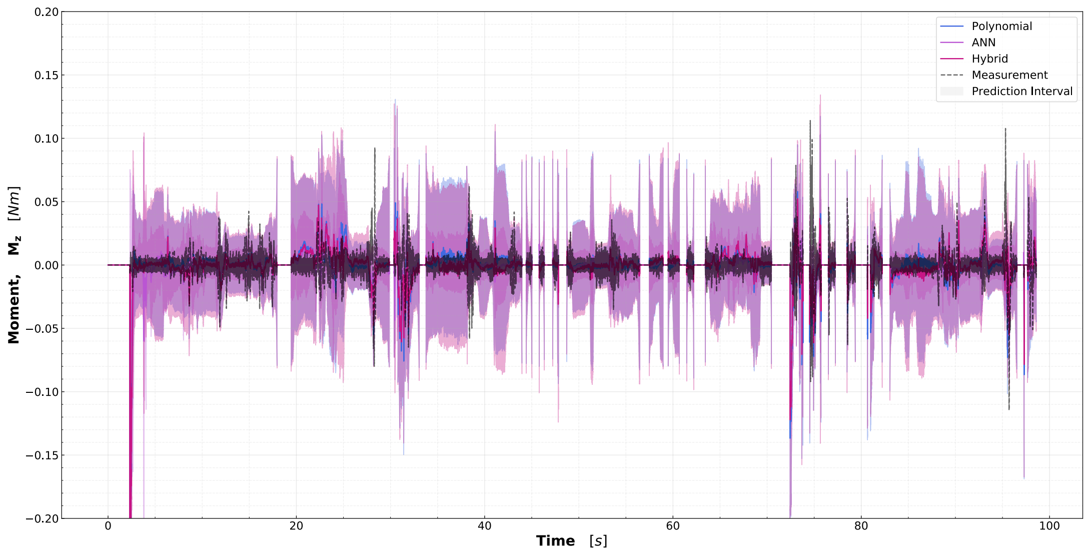


Fig. 74 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_z for flight 35 (outdoor untrained flight) of the MetalBeetle. Also shown is the measured M_z response (black dotted line). This flight was an line-of-sight outdoor flight in windy conditions (wind speed $\approx 6\ ms^{-1}$ with gusts of up to $\approx 10\ ms^{-1}$). The yellow highlighted region denotes backward flight away from the wind while the green highlighted region denotes forward flight into the wind.

5. *Identified polynomial model structures*

In this subsection, the regressors of the identified polynomial models of the outdoor MetalBeetle are summarized. Descending down the tables gives the order of selection and in grey are the fixed regressors. Also shown are the associated covariances and cumulative accuracy contribution of each regressor.

Table 5 Identified polynomial model of C_x for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	**Coefficient**	**Covariance**	**R^2**
$bias$	5.061E-02	0.0162	0.0000
$\bar{u}$	6.253E-02	0.0346	0.0016
$\sin(\theta)\bar{v}_{in}^2$	-1.008E-05	0.0000	0.0022
$\cos(\theta)\bar{\omega}_{tot}^4$	3.727E+00	36.6085	0.1806
$\bar{\omega}_{tot}\bar{q}$	1.184E+02	1.4878	0.2745
$\sin(\theta)\mu_{v_{in}}\bar{\omega}_{tot}$	1.947E+00	0.0165	0.3349
$\bar{q}$	-2.461E+01	0.5318	0.3618
$\bar{\omega}_{tot}\bar{u}\|\bar{v}\|^3$	-3.886E+00	0.1027	0.3811
$\bar{\omega}_{tot}^4$	-7.382E+01	1.3368	0.3960
$\sin(\psi)\mu_{v_{in}}\bar{\omega}_{tot}^3$	4.445E+00	0.1915	0.4067
$\sin(\theta)\mu_{v_{in}}^3\bar{\omega}_{tot}$	-1.398E-01	0.0058	0.4168
$\bar{\omega}_{tot}\bar{u}\|\bar{v}\|^2\bar{w}$	-7.406E+00	0.3267	0.4290
$\bar{\omega}_{tot}\bar{u}_q^3$	6.817E-01	0.0407	0.4380

Table 6 Identified polynomial model of C_y for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	-1.367E-02	0.0260	0.0000
$\bar{v}$	4.170E-01	0.0278	0.0088
$\sin(\phi)\bar{v}_{in}^2$	8.875E-06	0.0003	0.0096
$\sin(\phi)\bar{\omega}_{tot}$	-3.731E+00	0.0117	0.2612
$\bar{\omega}_{tot}\bar{p}$	-1.815E+02	2.4921	0.3453
$\cos(\psi)\mu_{v_{in}}^2\bar{\omega}_{tot}^2$	6.899E-01	0.0146	0.4009
$\bar{\omega}_{tot}\bar{p}^3$	5.308E+03	190.6813	0.4240
$\bar{\omega}_{tot}\bar{u}_p\lvert\bar{u}_r\rvert$	2.289E+00	0.0707	0.4399
$\bar{p}$	3.335E+01	0.7794	0.4534
$\bar{p}^3$	-1.072E+03	50.5911	0.4688
$\bar{\omega}_{tot}\lvert\bar{u}\rvert^3\bar{v}$	-4.724E+00	0.1536	0.4790
$\bar{v}^3$	-5.141E-01	0.0329	0.4894
$\sin(\phi)\lvert\bar{u}\rvert^4$	3.824E-01	0.0275	0.4978

Table 7 Identified polynomial model of C_z for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	3.218E-02	0.1640	-0.0000
$\bar{\omega}_{tot}^2$	-3.457E+00	0.9322	0.6187
$(\bar{u}^2 + \bar{v}^2)$	-1.112E-01	0.0535	0.6192
$(\bar{v}_{in} - \bar{w})^2$	1.451E-05	0.0000	0.6210
$\bar{w}$	2.078E-02	0.0765	0.6210
$\cos(\phi)\mu_{v_{in}}\bar{\omega}_{tot}$	8.373E-01	0.0545	0.7424
$\bar{\omega}_{tot}^4$	2.970E+02	0.4999	0.7759
$\bar{\omega}_{tot}\lvert\bar{q}\rvert$	2.669E+01	0.3324	0.7993
$\cos(\phi)\bar{\omega}_{tot}^2$	-1.361E+01	0.1009	0.8130
$\sin(\theta)\bar{\omega}_{tot}^3$	-1.574E+01	0.2042	0.8227
$\bar{\omega}_{tot}\lvert\bar{u}_r\rvert$	6.850E+00	0.1481	0.8315
$\cos(\theta)\mu_{v_{in}}$	-1.583E-01	0.0092	0.8383
$\cos(\psi)\lvert\bar{v}\rvert$	2.415E-01	0.0053	0.8434
$\cos(\theta)\bar{v}_{in}\bar{\omega}_{tot}^3$	-3.915E-01	0.0140	0.8471
$\lvert\bar{u}_r\rvert$	-9.813E-01	0.0455	0.8500

Table 8 Identified polynomial model of C_l for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	**Coefficient**	**Covariance**	**R^2**
$bias$	-1.568E-05	0.0000	0.0000
$\bar{p}$	-5.813E-03	0.0032	0.0002
$\bar{u}_p$	-3.920E-03	0.0001	0.3092
$\bar{\omega}_{tot}\bar{u}_p$	2.583E-02	0.0001	0.5848
$\bar{\omega}_{tot}\bar{u}_p^3$	3.252E-03	0.0001	0.6202
$\bar{\omega}_{tot}\bar{u}_p\|\bar{u}_r\|$	-7.680E-03	0.0002	0.6379
$\cos(\phi)\bar{u}_p\|\bar{u}_r\|^2$	6.547E-04	0.0000	0.6522
$\cos(\psi)\bar{\omega}_{tot}^2$	-9.156E-04	0.0001	0.6615
$\sin(\phi)\bar{\omega}_{tot}^4$	-6.530E-02	0.0041	0.6685
$\sin(\phi)\mu_{v_{in}}\bar{\omega}_{tot}^2$	3.799E-03	0.0003	0.6750
$\sin(\phi)\bar{u}_p^2$	-6.565E-04	0.0002	0.6781
$\bar{\omega}_{tot}\bar{p}$	2.834E-02	0.0094	0.6808
$\sin(\phi)\mu_z^2\mu_{v_{in}}$	-8.341E-05	0.0000	0.6834

Table 9 Identified polynomial model of C_m for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	**Coefficient**	**Covariance**	**R^2**
$bias$	6.022E-05	0.0000	0.0000
$\bar{q}$	-2.916E-03	0.0010	0.0001
$\bar{u}_q$	1.844E-03	0.0000	0.2389
$\bar{\omega}_{tot}\bar{u}_q$	-1.022E-02	0.0001	0.3262
$\bar{\omega}_{tot}\bar{u}_q^3$	-1.356E-03	0.0001	0.3741
$\sin(\theta)\mu_{v_{in}}\bar{\omega}_{tot}^3$	4.021E-03	0.0004	0.3851
$\bar{\omega}_{tot}\bar{q}^3$	8.806E-01	0.0570	0.3974
$\bar{q}\|\bar{v}\|\bar{w}^2$	2.385E-02	0.0036	0.4073
$\cos(\psi)\bar{\omega}_{tot}^3$	-3.659E-03	0.0006	0.4149
$\bar{u}_q\|\bar{v}\|^3\bar{w}$	-1.195E-03	0.0002	0.4230
$\sin(\psi))\bar{u}^2\|\bar{v}\|^2$	-1.005E-03	0.0002	0.4301
$\bar{\omega}_{tot}\bar{q}^3\|\bar{r}\|$	-7.923E+01	19.7461	0.4352
$\sin(\theta)\bar{u}^2$	-3.601E-04	0.0001	0.4404

Table 10 Identified polynomial model of C_n for the MetalMeetle (Outdoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	1.905E-05	0.0000	0.0000
$\bar{r}$	-2.196E-03	0.0010	0.0016
$\bar{u}_r$	5.916E-04	0.0000	0.1255
$\bar{\omega}_{tot}\bar{u}_r$	-3.787E-03	0.0001	0.2045
$\bar{\omega}_{tot}\lvert\bar{u}_q\rvert^2\lvert\bar{u}_p\rvert\bar{u}_r$	1.559E-03	0.0002	0.2264
$\bar{\omega}_{tot}\bar{u}_r^3$	-3.072E-04	0.0000	0.2401
$\sin(\psi)\lvert\bar{u}_q\rvert\lvert\bar{u}_p\rvert^3$	-9.452E-04	0.0001	0.2522
$\lvert\bar{u}_q\rvert\lvert\bar{u}_p\rvert^3$	4.686E-04	0.0001	0.2610
$\bar{u}_p\lvert\bar{v}\rvert^2$	-6.710E-04	0.0001	0.2707
$\bar{\omega}_{tot}\lvert\bar{p}\rvert\bar{r}$	-2.672E-01	0.0576	0.2772
$\bar{u}_p\lvert\mu_y\rvert^2\mu_{v_{in}}^2$	1.352E-05	0.0000	0.2829
$\lvert\bar{p}\rvert\lvert\bar{v}\rvert^3\bar{w}$	-1.891E-02	0.0026	0.2866
$\lvert\bar{p}\rvert\lvert\bar{v}\rvert\bar{w}^3$	2.193E-02	0.0051	0.2919

B. Detailed hovering simulation results of identified MetalBeetle models

The identified models of the outdoor MetalBeetle are placed in the quadrotor simulation to evaluate their responses to hovering flight. The hovering flight is used to modify the PID controllers and provides for a baseline understanding of the models' performances. The subsequent simulation results of the identified polynomial-only, hybrid, and ANN-only outdoor models of the MetalBeetle are summarized here.

1. Polynomial model of the MetalBeetle

The rotor inputs and subsequent attitude response of the polynomial model of the outdoor MetalBeetle for hovering flight are illustrated in fig. 75 and fig. 76 respectively. Surprisingly, despite the poor model performances and contamination of wind, the polynomial model appears to be able to maintain a near-idle rotor speed with attitudes tending to zero. Collectively, these results suggest that the model is stabilized.

However, instabilities are apparent in the force responses of the quadrotor, depicted in fig. 77. This is expected due to the contamination of wind. The F_x model exhibits a non-zero force during hover while all other forces realistically tend to zero. Various trimmed flight conditions, with positive and negative attitudes, of the MetalBeetle were also conducted to investigate the effects on this constant force. However, no matter the trimmed condition, this constant term persisted at some point in the simulation. For example, the initial forces may be around zero but after some time the F_x would slowly converge back to this negative bias. It is found that the term, $\bar{\omega}_{tot}^4$ introduces this non-zero force in F_x (see fig. 79). This explains the persistence of this force bias, given the sole dependence on the rotor speed and apparent invariance of the velocity. Recall from the polynomial structures that yaw - and thus wind - is introduced to F_x model through $\sin(\psi)$ and F_y model through $\cos(\psi)$. In the hover condition, $\psi = 0$, and thus the F_x model may lack this wind correction term and subsequent experiences a force bias.

Conversely, the corresponding moment responses do not appear to harbor any instabilities in the hovering condition. These responses are shown in fig. 78. Aside from small oscillations at the beginning of the response associated with stabilizing attitude corrections, these moments exhibit the expected behaviour as they are equal to zero during hover. The associated PIs are compelling as they imply a lower confidence for the pitching moment, M_y, model, which is associated with the non-zero F_x force. Although the M_y $MPIW$s are found to be wider than M_x, they are similar to those of M_z. Hence, there may be a link between the moment confidence and the force discrepancies or, more broadly, direction of incident wind. For all models, the PIs during hover are generally large and reflect the uncertainty in this

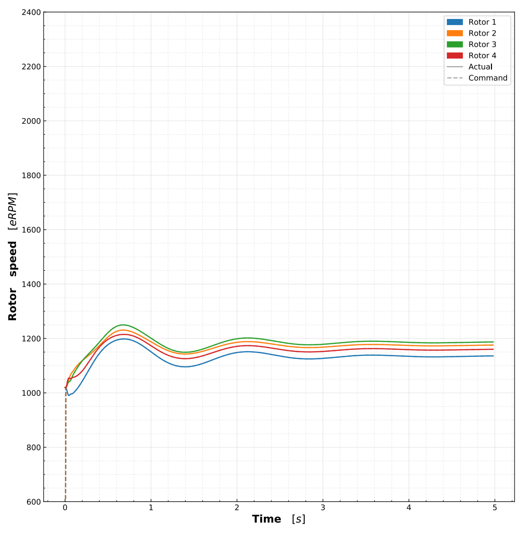


Fig. 75 Commanded rotor speed of the identified outdoor polynomial model of the MetalBeetle during hovering flight.

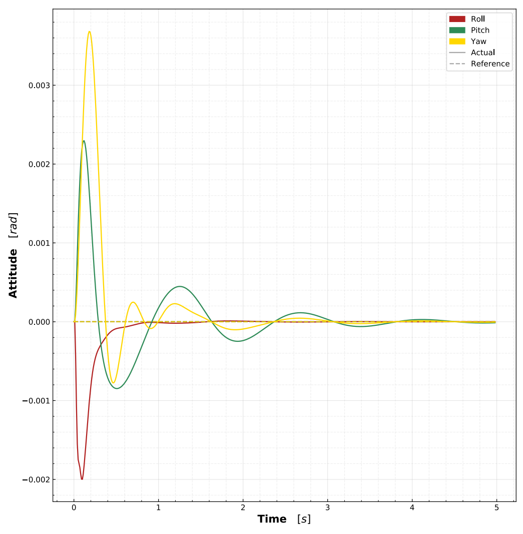


Fig. 76 Attitude response of the identified outdoor polynomial model of the MetalBeetle during hovering flight.

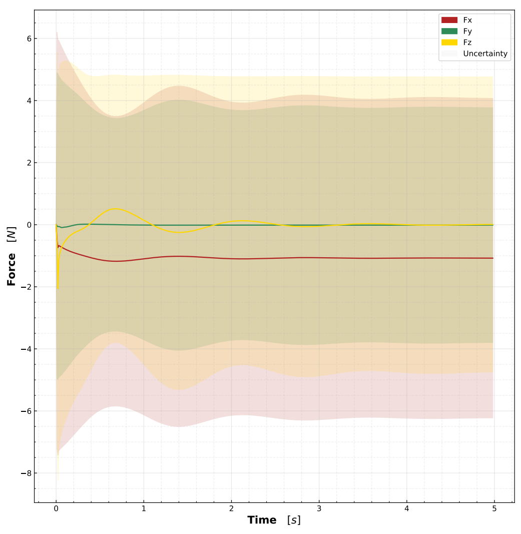


Fig. 77 Force response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle during hovering flight.

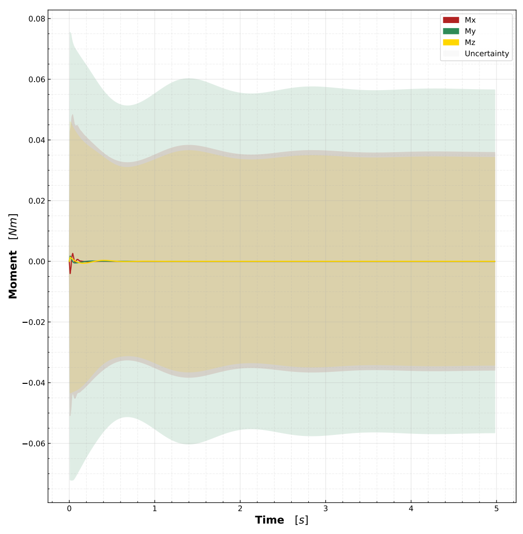


Fig. 78 Moment response, along with associated prediction intervals, of the identified outdoor polynomial model of the MetalBeetle during hovering flight.

condition - which is outside the envelope of the training data.

2. Hybrid model of the MetalBeetle

With respect to the hovering flight simulation of the outdoor MetalBeetle hybrid model, fig. 80 and fig. 81 depict the commanded rotor speeds and subsequent attitude respectively. It is clear from these responses alone that the outdoor MetalBeetle hybrid model suffers stability. A peculiar fanning-out of the commanded rotor speeds occurs. This is in contrast to reality where such rotor speeds should be similar, as in the polynomial simulation. The expansion of the rotor speeds is suspected to relate to predictions made by the ANN component using the runaway velocity arising from the non-zero F_x. The states, and combinations therein, are likely unfamiliar to the ANNs which eventually leads to

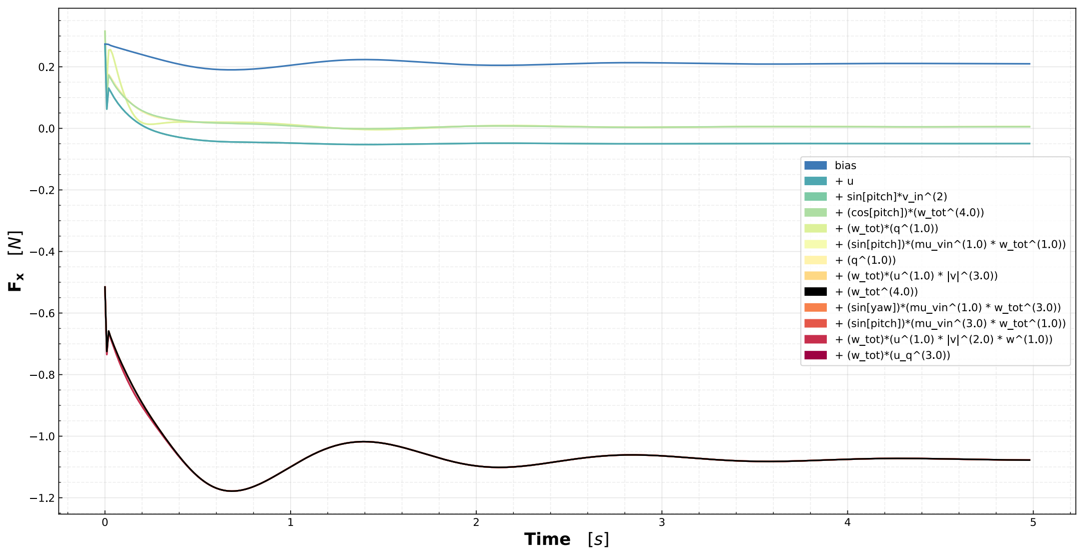


Fig. 79 Cumulative contributions of the constituent regressors of the identified polynomial model of F_x for the outdoor MetalBeetle during hovering flight. Highlighted in black is the regressor which adds the negative asymmetry due to wind to the model. Regressor contributions are in descending order in the legend.

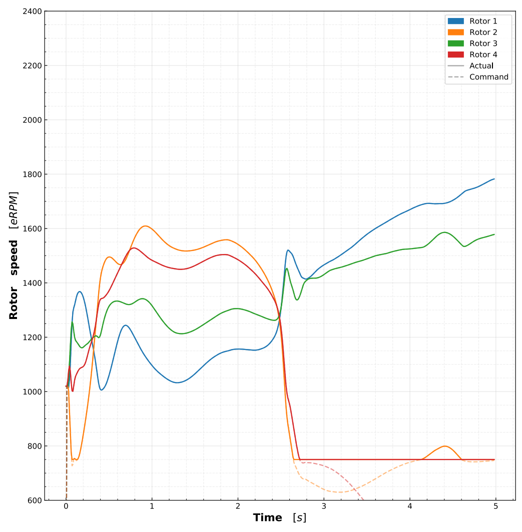


Fig. 80 Commanded rotor speed of the identified outdoor hybrid model of the MetalBeetle during hovering flight.

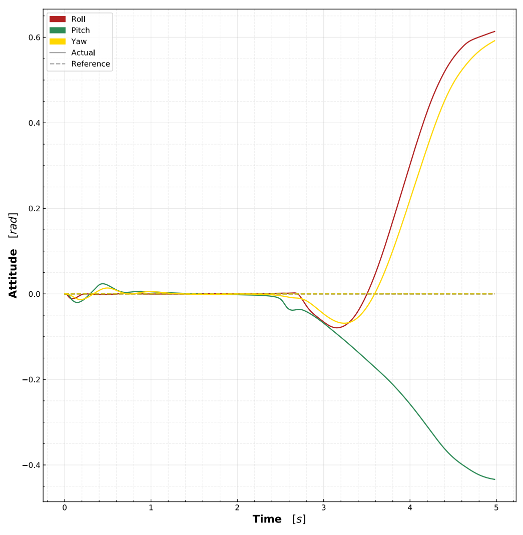


Fig. 81 Attitude response of the identified outdoor hybrid model of the MetalBeetle during hovering flight.

these unpredictable outputs across all force and moment models given the dense nature of the ANN structure. The moments in particular influence the necessary control response by the rotor speeds. Indeed, the instability of the hybrid model is evident through the rapid change in rotor commands at around 2.6 seconds instigated by the slowly diverging attitude. Regardless of how the controller responds, once these rotor speeds hit saturation, the attitudes explode and the quadrotor tends towards instability.

The corresponding force and moment responses of the hovering simulation are illustrated in fig. 82 and fig. 83 respectively. Both the force and moment responses also exhibit non-zero responses at the start of the simulation due to the stabilization of the quadrotor from its moment of inertia asymmetries. Comparing the subsequent responses

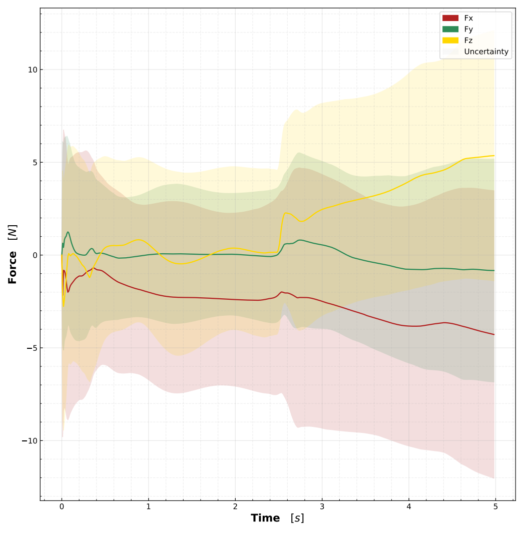


Fig. 82 Force response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle during hovering flight.

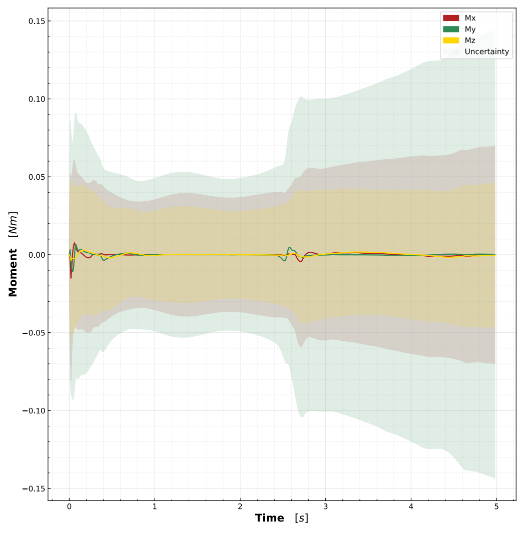


Fig. 83 Moment response, along with associated prediction intervals, of the identified outdoor hybrid model of the MetalBeetle during hovering flight.

with the polynomial equivalents reveals the minute modifications made by the ANN compensator. Unfortunately, the ANN compensator is not able to remove the F_x force bias suspected to be a consequence of unknown wind. The force responses resemble their polynomial counterparts until the moment of rotor saturation, wherein all forces begin increasing in magnitude due to the non-zero attitude angles. These attitude changes originate from small excitations in moment responses that occur at around 2.5 seconds. This excitation likely derives from the ANN components since this behavior is not observed in underlying polynomial model. Furthermore, the saturation of the commanded rotors induces an increase in PI widths across all models, which is indicative of the greater uncertainty with the associated predictions and highlights the utility of these intervals.

3. ANN model of the MetalBeetle

The simulation results of the identified ANN outdoor MetalBeetle models indicate an unrealistic and invalid model, particularly due to a poor F_z model. The effects of this erroneous model are apparent in the hovering flight simulations.

The commanded rotor speeds of the identified ANN outdoor MetalBeetle models during hovering flight are shown in fig. 84. The characteristic fanning of the commanded rotor speeds for identified ANN models are once again present and eventually lead to instability due to rotor saturation. Again, this expansion of the rotor speeds is suspected to arise from the ANN moment predictions on unfamiliar inputs which induce gradual attitude drifts. While the subsequent saturation of the rotor speeds does not immediately induce instability in the associated attitude response, illustrated in fig. 85, they initiate the divergence of these attitudes from zero. Soon after, the attitude responses blow up and the controller struggles to maintain control with rapidly oscillating commanded rotor speeds.

The associated force response during hover is depicted in fig. 86. Indeed, all forces appear to harbor biases and lead to runaway terms. For F_x and F_y, these likely originate from the influence of unknown wind while for F_z the exact causes are unknown. The F_z ANN model unrealistically represents the thrust force. Figure 88 depicts the thrust profile of the ANN-only, polynomial-only, and hybrid models as a function of the average rotor speed with all other inputs set to zero. The ANN F_z model predicts forces that are consistently above zero (i.e. hover thrust). As such, there is no way for the model to descend, which induces many of the stability issues. There appears to be a clear difficulty for the ANN models to capture the F_z forces for the outdoor models. Perhaps this lies with the asymmetry of the thrust data but may also stem from an over-reliance on the total rotor speed, which is always positive. Future work could therefore investigate alternative normalization schemes which include negative terms, such as a correction first for the hovering rotor speed. Indeed, the velocity along w encodes this information, but in outdoor flight this is obtained from the GPS data, which does not accurately capture the velocities as seen in fig. 40 and fig. 41. In any case, all the force models are invalid and promote runaway velocities as shown in fig. 89. These runaway velocities further promote unstable

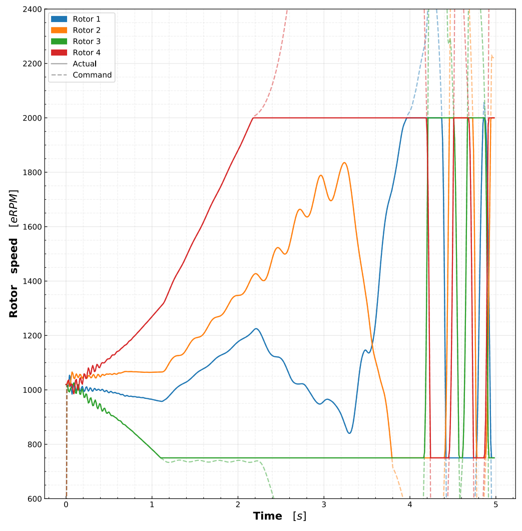


Fig. 84 Commanded rotor speed of the identified outdoor ANN model of the MetalBeetle during hovering flight.

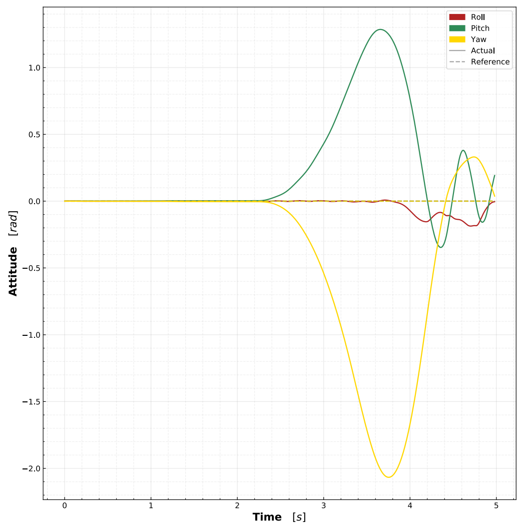


Fig. 85 Attitude response of the identified outdoor ANN model of the MetalBeetle during hovering flight.

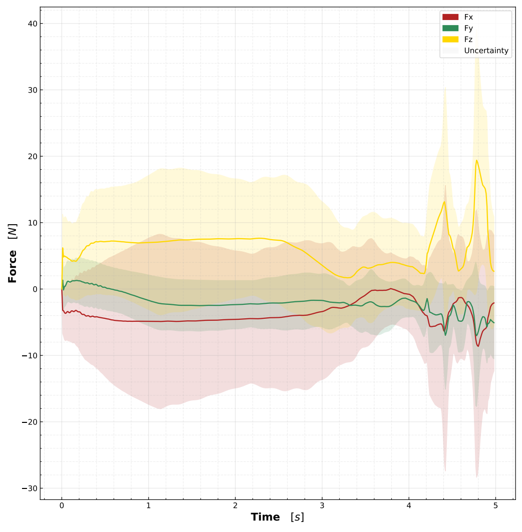


Fig. 86 Force response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle during hovering flight.

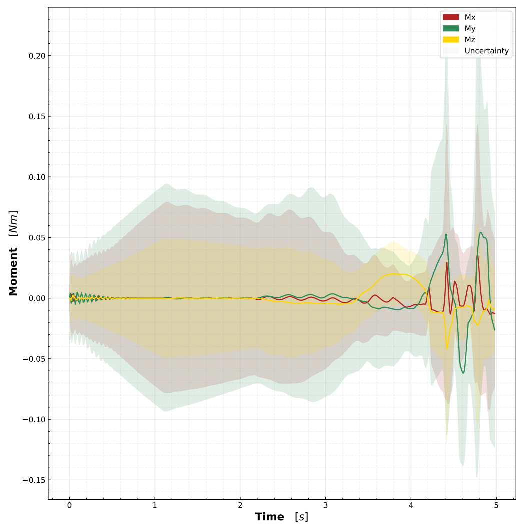


Fig. 87 Moment response, along with associated prediction intervals, of the identified outdoor ANN model of the MetalBeetle during hovering flight.

predictions by the ANNs. Fortunately, the associated PIs indicate the lack of confidence in the model predictions, demonstrating their utility. Interestingly, these PIs first grow as the commanded rotor speeds near saturation, perhaps forecasting this instability.

Figure 87 illustrates the moment responses of the identified ANN outdoor MetalBeetle model in simulated hovering flight. Rapid oscillations at the beginning of the moment responses correspond to the initial stabilization efforts of the quadrotor. However, these oscillations begin to grow again quickly after controller manages to stabilize the quadrotor. As these oscillations are much lower in frequency, they likely stem from predictions made on the runaway terms and thus propagate the instabilities. Again, the PIs appear to forecast this behavior as they widen before the low-frequency oscillations start.

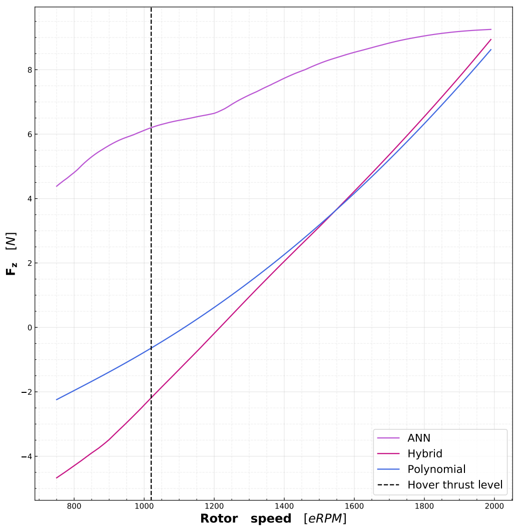


Fig. 88 Thrust profile, as a function of the average rotor speed, of the identified polynomial-only, ANN-only, and Hybrid models of the outdoor MetalBeetle. Here, zero corresponds to the thrust necessary to maintain hover. As such, also shown is the true hovering rotor speed of the MetalBeetle. Note that this profile is generated with all other states set to zero.

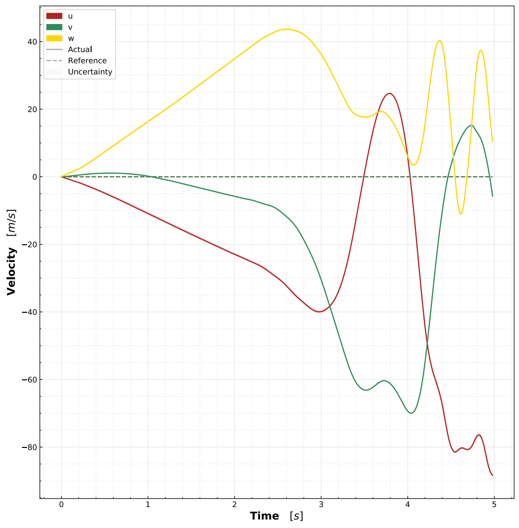


Fig. 89 Velocity response of the identified outdoor ANN model of the MetalBeetle during hovering flight.

C. Outdoor MetalBeetle simulation PID gains

The PID gains used for the controllers in the quadrotor simulation are summarized here.

1. Polynomial model of the MetalBeetle

Associated PID gains for rate controller - transforming rate command into rotor speeds.

$$P_{rate} = \begin{bmatrix} 10 & 10 & 10 \\ 10 & 10 & 10 \\ 10 & 10 & 10 \\ 10 & 10 & 10 \end{bmatrix}, \quad I_{rate} = \begin{bmatrix} 100 & 100 & 100 \\ 100 & 100 & 100 \\ 100 & 100 & 100 \\ 100 & 100 & 100 \end{bmatrix}, \quad D_{rate} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 2 & 2 \end{bmatrix} \tag{74}$$

Associated PID gains for attitude controller - transforming attitude command into rate command.

$$P_{att} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{att} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad D_{att} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{75}$$

Associated PID gains for altitude controller - transforming vertical velocity command in earth frame to rotor speeds.

$$P_{alt} = \begin{bmatrix} 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad D_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{76}$$

2. Hybrid model of the MetalBeetle

Associated PID gains for rate controller - transforming rate command into rotor speeds.

$$P_{rate} = \begin{bmatrix} 10 & 10 & 10 \\ 10 & 10 & 10 \\ 10 & 10 & 10 \\ 10 & 10 & 10 \end{bmatrix}, \quad I_{rate} = \begin{bmatrix} 100 & 100 & 100 \\ 100 & 100 & 100 \\ 100 & 100 & 100 \\ 100 & 100 & 100 \end{bmatrix}, \quad D_{rate} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 2 & 2 \end{bmatrix} \tag{77}$$

Associated PID gains for attitude controller - transforming attitude command into rate command.

$$P_{att} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{att} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad D_{att} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1.3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{78}$$

Associated PID gains for altitude controller - transforming vertical velocity command in earth frame to rotor speeds.

$$P_{alt} = \begin{bmatrix} 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad D_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{79}$$

3. ANN model of the MetalBeetle

Associated PID gains for rate controller - transforming rate command into rotor speeds.

$$P_{rate} = \begin{bmatrix} 30 & 40 & 30 \\ 30 & 40 & 30 \\ 30 & 40 & 30 \\ 30 & 40 & 30 \end{bmatrix}, \quad I_{rate} = \begin{bmatrix} 250 & 500 & 250 \\ 250 & 500 & 250 \\ 250 & 500 & 250 \\ 250 & 500 & 250 \end{bmatrix}, \quad D_{rate} = \begin{bmatrix} 1.5 & 2 & 4 \\ 1.5 & 2 & 4 \\ 1.5 & 2 & 4 \\ 1.5 & 2 & 4 \end{bmatrix} \tag{80}$$

Associated PID gains for attitude controller - transforming attitude command into rate command.

$$P_{att} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{att} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 100 \end{bmatrix}, \quad D_{att} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \tag{81}$$

Associated PID gains for altitude controller - transforming vertical velocity command in earth frame to rotor speeds.

$$P_{alt} = \begin{bmatrix} 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \\ 0 & 0 & 10 \end{bmatrix}, \quad I_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad D_{alt} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{82}$$

D. Supplementary model identification: Indoor MetalBeetle

Although not the focus of the present paper, indoor models of the MetalBeetle are also identified to provide a reference of comparison of the corresponding outdoor models.

1. Indoor flight data acquisition

Before the transition to outdoor flight, indoor flights are used to verify the functionality of the pipeline and to design input manoeuvres that adequately excite the forces and moments along each of the axes. However, data gathered indoors will be constrained to the low-speed domain of the flight envelope.

The indoor flights were conducted at the CyberZoo, located at the faculty of Aerospace Engineering at the Delft University of Technology (TU Delft). This facility is confined to a space of approximately 7 m tall, 10 m wide, and 10 m long. The CyberZoo is equipped with an external OptiTrack motion capturing system which records the quadrotor's attitude and position at 120 Hz by tracking the IR markers attached to the quadrotor (e.g. see IR rig attached to the MetalBeetle fig. 2). These data files are stored on the computer connected to the OptiTrack system. The quadrotor's velocity is then derived from the evolution of its tracked position over time. Therefore, the OptiTrack system is analogous to the role of the GPS module for outdoor flights.

The quadrotor's acceleration, attitude, and rotational rates are measured by the on-board IMU and logged at 500 Hz. The Electronic Speed Controller (ESC) measures the rotational rate of each of the propellers in eRPM (electrical RPM), which is a proxy measure for the true RPM of the quadrotor. As with the IMU information, the rotor speeds are logged by on-board the quadrotor at 500 Hz.

Both LOS and FPV flights were conducted indoors and, prior to take-off, the quadrotor was switched on and held stationary for a few seconds to facilitate the estimation of the sensor biases during raw data processing.

As with the outdoor flight, the LOS flights are intended to excite the dynamics along each of the axes in a, somewhat, repeatable manner to have consistent excitations across some of the flights. Therefore, these manoeuvres are, in principle, the same as those conducted outdoors but confied to the space of the CyberZoo. An example of the flight trajectory during this z-axis manoeuvre can be seen in fig. 90. Also apparent in this figure are the bounds of the Cyberzoo tracking through the flat line wherein the quadrotor exceeds the OptiTrack field-of-view.

These manoeuvres, however, excite the forces more vigorously than the moments. Therefore, to entice moment excitations, rapid oscillations in the roll, pitch and yaw were conducted which entailed rapidly moving between the extremes of the angular commands. In essence, this is a higher-frequency version of the sinusoidal step inputs which does not allow the quadrotor to translate as much and excites the moments more often. As the yaw excitations are separate from the z-axis manoeuvres, ramp inputs were given to the yaw angle to induce a spinning of the quadrotor about its yaw axis. Again, these manoeuvres are spread over, and appear in, multiple flights.

However, it is difficult to reach high velocities and perform aggressive manoeuvres for such LOS flight indoors. Instead, FPV flights are used to achieve higher velocities by flying in circles (both counter-clockwise and clockwise) around a flag placed at the center of the CyberZoo with the aim of flying as fast as possible. An example of this flight is depicted in fig. 91. While perhaps paradoxical for indoor flights, relatively high velocities, up to 5 ms^{-1}, were achieved

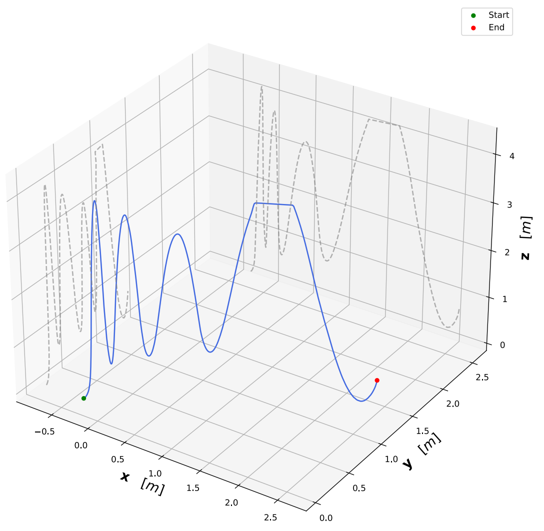


Fig. 90 Illustrative example of the flight trajectory during an LOS z-axis up-and-down manoeuvre. Also shown, in grey, are the two-dimensional traces of this flight in the $x - z$ and $y - z$ planes for reference.

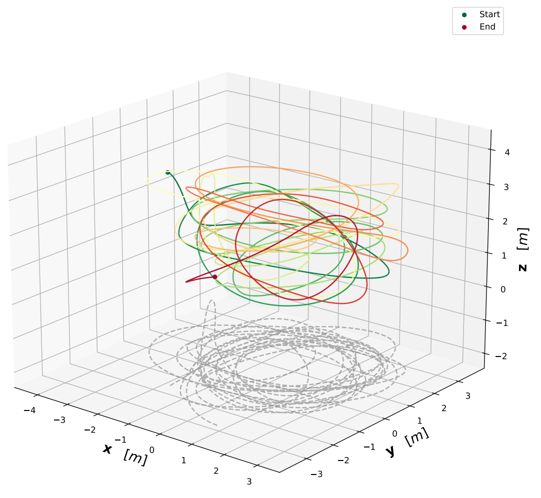


Fig. 91 Illustrative example of the flight trajectory during an FPV flight. Also shown, in grey, is the ground track of this flight. Time is encoded through color progressing from green to red.

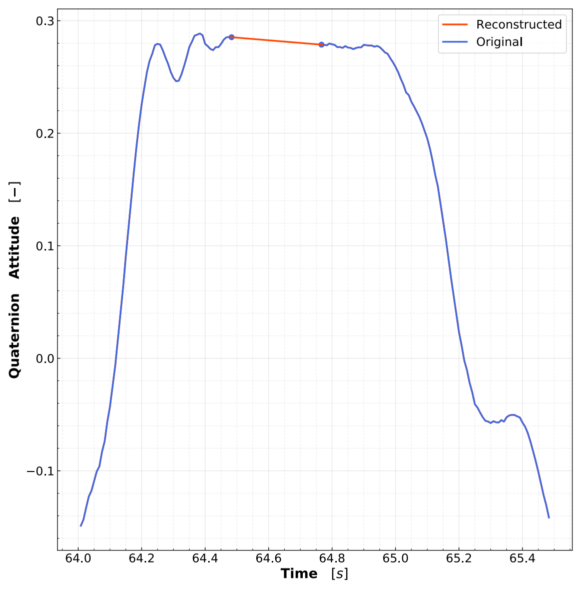


Fig. 92 Reconstruction of missing OptiTrack measurements (in blue) through a linear spline interpolation (in orange). Here, the interpolation is used to reconstruct estimations of the MetalBeetle's attitude.

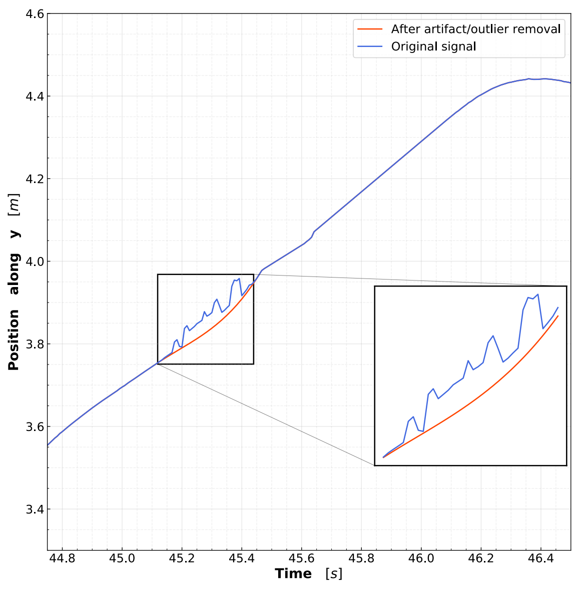


Fig. 93 Illustrative example of results of the outlier and artefact removal algorithm, as applied from the OptiTrack measurements of the MetalBeetle's position. A cubic spline is employed to remove the artefacts.

in this manner.

Since the indoor experiments rely on an external motion capturing system to facilitate the velocity and position estimations, additional pre-processing is required to filter, align, and synchronize the OptiTrack-derived data with the on-board IMU data. Such processing is not required in the outdoor experiments since all relevant states are measured through on-board sensors and synchronized through the BetaFlight logging module.

Unfortunately, the OptiTrack system would occasionally lose tracking of the quadrotor either due to poor visibility of the markers or the flight of the quadrotor beyond the field-of-view of the OptiTrack camera system (refer to fig. 90. The loss-of-tracking events culminated in missing data and discontinuities in tracking. Missing data is reconstructed through a linear spline interpolation. Linear splines were employed over cubic splines since attempting to interpolate such regions with a cubic spline resulted in unwanted, and unrealistic, ringing artifacts. Consider, for example, an ascent-descent manoeuvre for which the quadrotor leaves the trackable space. As it leaves, it holds an upwards velocity whereas, when it enters again, it does so with a downwards velocity. The ringing artifacts occur when attempting to interpolate this step-like signal. While imperfect, the linear interpolation reproduces a more faithful reconstruction of the true states. An example of this interpolation, taken from one of the MetalBeetle's flights, is given in fig. 92 and highlights the step-like nature of some of these signals.

Moreover, these discontinuities and tracking issues contaminate the OptiTrack measurements with outliers and artefacts. To detect such outliers and artifacts, the numerical derivative of the OptiTrack data was taken and resultant absolute magnitudes evaluated. Magnitudes which were larger than six standard deviations of the mean magnitude were considered to be an artifact or outlier. This procedure is consistent with the processing pipeline of Sun et al. [8]. Subsequently, the artifact or outlier points - along with their immediately surrounding points ($N = 10$) - were removed from the data. This is done to ensure that the base of the outlier peak is also removed. The subsequent gap in data was then reconstructed using a cubic spline interpolation. Figure 93 demonstrates how this algorithm removes artefacts present in the position measurements of one of the MetalBeetle's flights. While imperfect at removing all oscillations, the most prominent artefacts are removed as illustrated by the highlighted region in fig. 93. Moreover, since the surrounding points were also removed, this interpolation did not produce aforementioned strong ringing artifacts.

The OptiTrack motion capturing system records data with respect to its own local OptiTrack axes system. Note that, in reality, the OptiTrack system does not directly track the position and orientation of the quadrotor itself, but rather the position and orientation of the markers affixed to the quadrotor. For the MetalBeetle, these markers are attached to

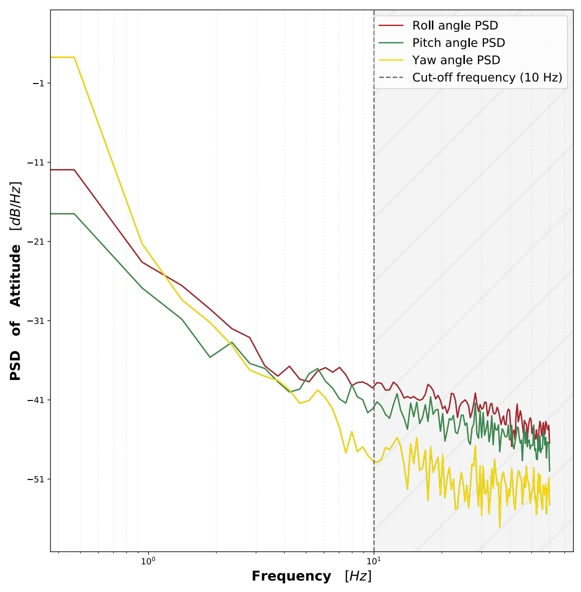


Fig. 94 Power spectral density of the OptiTrack measurements of the attitude of the DataCan75 in hovering flight. The grey-dotted line denotes the chosen cutoff frequency.

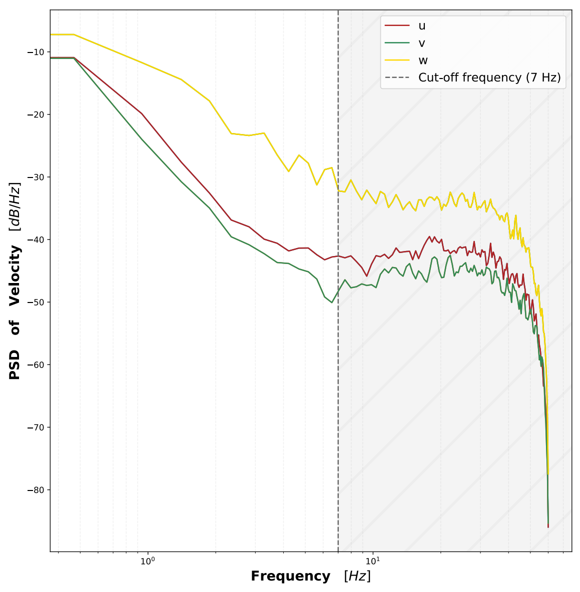


Fig. 95 Power spectral density of the OptiTrack-derived velocities, in $\{E\}$, from hovering flight of the DataCan75. The grey-dotted line denotes the chosen cutoff frequency.

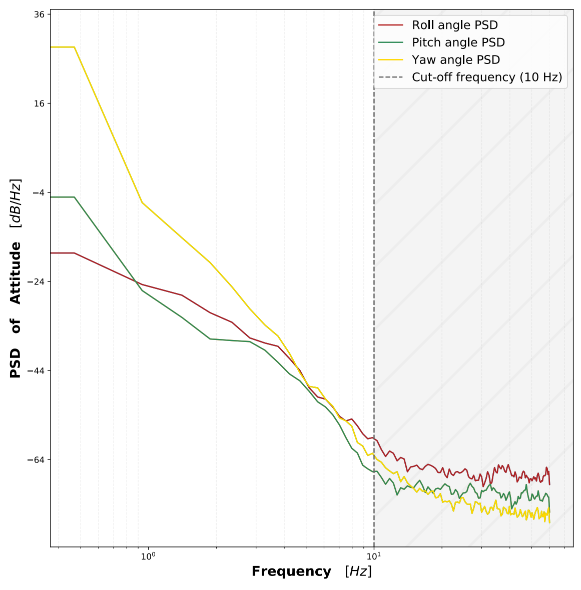


Fig. 96 Power spectral density of the OptiTrack measurements of the attitude of the MetalBeetle in hovering flight. The grey-dotted line denotes the chosen cutoff frequency.

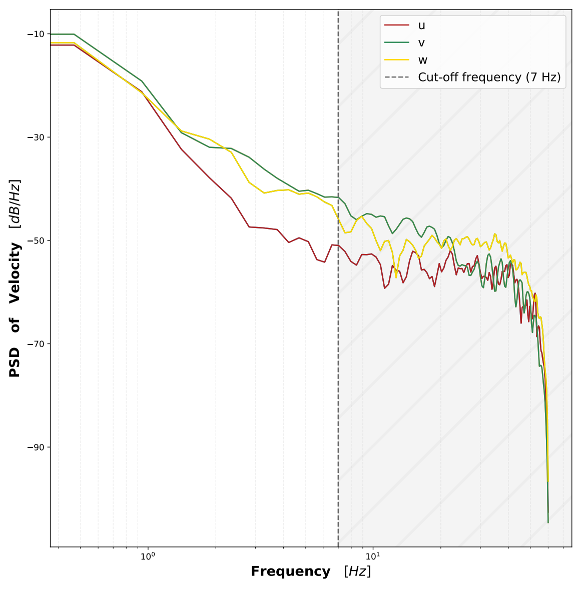


Fig. 97 Power spectral density of the OptiTrack-derived velocities, in $\{E\}$, from hovering flight of the MetalBeetle. The grey-dotted line denotes the chosen cutoff frequency.

an independent marker-rig which is strapped above the battery (refer to fig. 2) and are thus offset with respect to its center of rotation. Hence, where necessary, the OptiTrack data is then aligned with the quadrotor's geometric center (it is assumed that this coincides with the center of rotation). Indeed, the attachment of the OptiTrack markers alters the geometric properties of the quadrotor and thus influences the measured aerodynamic forces and moments. For the purposes of this paper, variations across flights (e.g. due to movement of the marker-rig during flight) are assumed to be negligible. Following the adjustment of the OptiTrack measurements to the quadrotor's geometric center, the OptiTrack-derived measurements are transformed from the OpitTrack local reference frame into the earth reference frame, $\{E\}$. The quadrotor's velocity is then obtained by taking the numerical derivative of the position estimates, in $\{E\}$, through central differences [8].

Even up to this point, there still remains some rapid oscillations in the OptiTrack measurements which originate from the difficulty in tracking the quadrotors. While not particularly an issue for the tracking of the Metaleetle (i.e. minimal oscillations), these tracking issues were prominent for smaller quadrotors, such as the DataCan75[¶¶¶], likely due to the close proximity of the tracking markers. Note that the quality of these IR markers significantly affects the tracking performance. For example, (tape) markers are used for the DataCan75 instead of the recommended tracking markers since the markers are too large to be fixed to the small quadrotor. This IR tape is harder for the system to track. The subsequent marker recognition and distinction issues - which are also seen as flickering in the OptiTrack system - occurred even while the DataCan75 was stationary (i.e. in hover). Consequently, for generalizability and robustness of the processing pipeline, both the OptiTrack derived attitude and velocity of the quadrotor were further filtered using a 4th order low-pass Butterworth filter with cutoff frequencies of 10 and 7 Hz respectively. These cutoff frequencies are chosen based on the Power Spectral Densities (PSDs) of the OptiTrack measurements of the attitude and position-derived velocity during hovering flight of the quadrotors. Hovering flight was used for this filter parameterization to best approximate the noise associated with the marker-recognition based tracking issues. Figures 94 and 95 respectively illustrate the PSDs of the attitude and velocity from this hovering flight for the DataCan75 and figs. 96 and 97 those for the MetalBeetle. Illustrative examples of this filtering for the roll angle, ϕ, and velocity along x_E, u, for an arbitrary (manoeuvred) flight of the DataCan75 are given by figs. 98 and 99 respectively.

The OptiTrack attitude is then further transformed to describe the rotation between $\{E\}$ and the quadrotor's body reference frame, $\{B\}$. Through this rotation matrix, the velocities may be transformed from $\{E\}$ to $\{B\}$ to complete the state vector.

However, there remains a sampling frequency discrepancy between the two datastreams. Subsequently, the OptiTrack data is up-sampled, through a cubic spline interpolation, from 120 Hz to 500 Hz to be consistent with the sampling rate of the on-board IMU. Note that the OptiTrack data is up-sampled, rather than the IMU data down-sampled, following

[¶¶¶]The DataCan75 is a TinyWhoop quadrotor also used to gather indoor flight data and evaluate the system identification pipeline. Even though it is not discussed further here, it is desirable that the pipeline is compatible with such quadrotors.

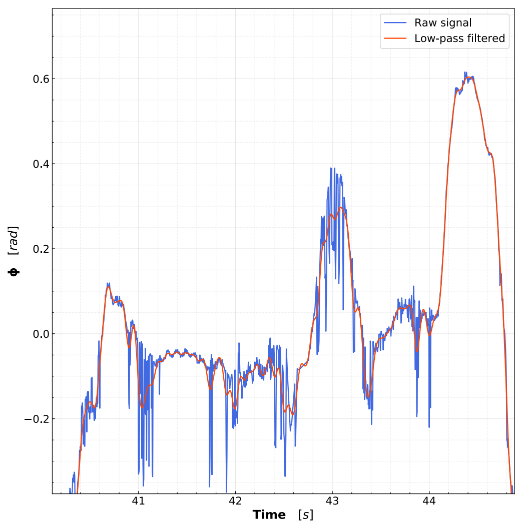


Fig. 98 Comparison of the OptiTrack measured roll angle, ϕ, from an arbitrary flight of the DataCan75 before (in blue) and after (in orange) the application of the low-pass filter.

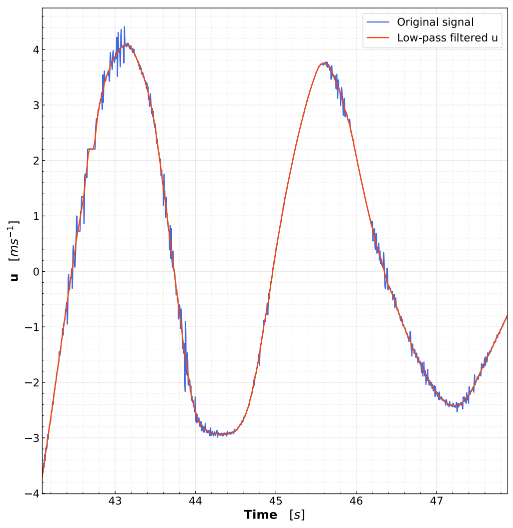


Fig. 99 Comparison of the OptiTrack-derived horizontal velocity, u, from an arbitrary flight of the DataCan75 before (in blue) and after (in orange) the application of the low-pass filter.

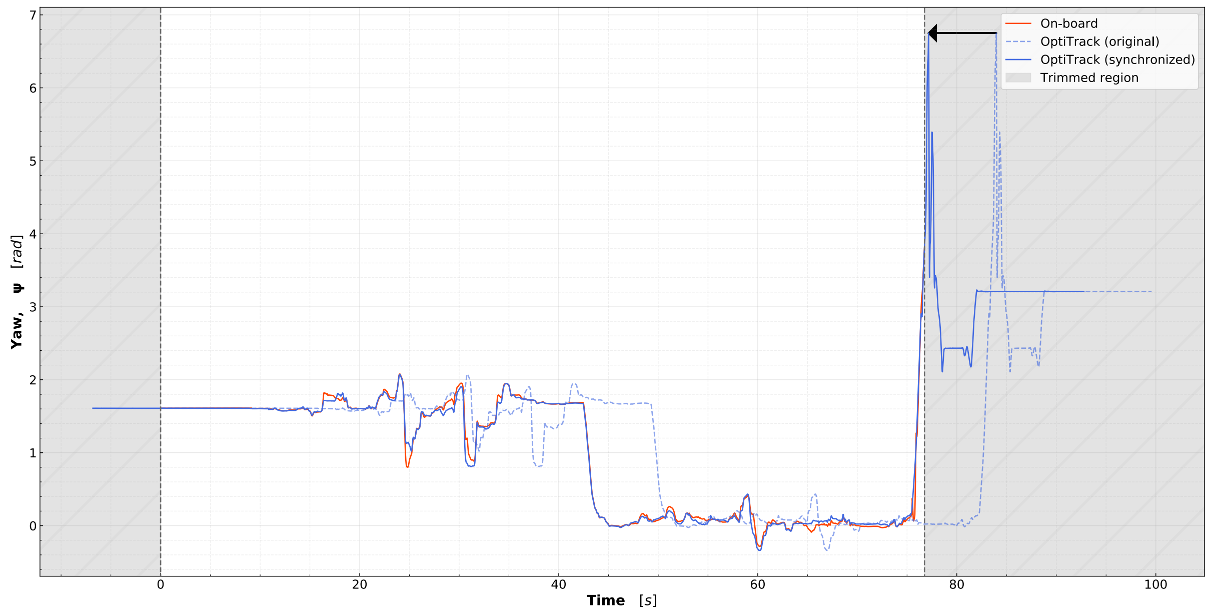


Fig. 100 Example alignment of the OpitTrack data with the on-board measurements and subsequent trimming of excess data. Here, the alignment of the yaw angle, ψ for a flight from the MetalBeetle is shown. The black arrow indicates the direction of delay shift applied to the OptiTrack signal

the data pre-processing procedure of Sun et al. [8] and to preserve information in the measurement data. Furthermore, to facilitate a fair comparison between indoor and outdoor models, resampling data to 500 Hz is in any case desirable since the outdoor models solely rely on measurements logged at 500 Hz. This allows for higher-frequency dynamics to be captured (e.g. related to the fast actuation of the rotors and the interactions therein).

As the states thus far have been recorded by two independent systems, there are likely delays between the two datastreams. Therefore, the OptiTrack data is synchronized with the on-board data through a cross-correlation of the attitude angles, which should largely be the consistent (i.e. aside from loss-of-tracking events). The delay between the two datastreams is taken as the lag which results in the largest cumulative positive correlation across all the attitude angles. This is done since the lag obtained from each attitude individually is slightly different. During the alignment of the two datastreams, the time arrays are truncated to the largest region for which there is data available from both the IMU and the OptiTrack system. Figure 100 illustrates the results of this alignment algorithm for the yaw angle, ψ, as applied to a flight from the MetalBeetle. Note that, for the final state vector, the attitude estimate is taken from the IMU and not the OptiTrack system. The motivation behind this stems from the fact that the IMU attitude is continuous (no loss-of-tracking events) and is the primary source of attitude information for the outdoor experiments.

Subsequently, the indoor data is processed following the general data processing procedure outlined in section IV.B.2.

2. *Identified indoor MetalBeetle models*

The performance metrics of the identified force and moment models of the MetalBeetle for indoor flight are summarized in table 11 and table 12 respectively. In these tables, the measures of accuracy ($NRMSE$ and R^2) and PI quality ($PICP$ and $MPIW$) are shown for both the entire flight data set and the testing data subsets (i.e. flights which are entirely unseen during training). The results discussed here will predominantly consider the performance of the indoor MetalBeetle models with respect to this testing data subset to better assess the validity and generalizability of the identified models.

The identified ANN F_x and F_y models of the indoor MetalBeetle show good model performance while the F_z model's performance is lackluster. The ANN model for F_x manages an acceptable test accuracy (R^2 = 0.8827) with almost valid and narrow PIs ($PICP$ = 94.3 < 95, $MPIW$ = 5.8%). Strictly speaking, the ANN F_x model of the indoor MetalBeetle does not satisfy the $PICP \geq 95$ criterion, although it is close to this condition. In the context of its higher PICP for the full data set ($PICP$ = 97.1% > 95%), the ANN F_x model is likely over-fitting the training

Table 11 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid force models for the MetalBeetle (Indoor flight). The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Fx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0142	0.0171	0.9056	0.8827	97.1247	94.2780	5.4779	5.7883
Poly	0.0151	0.0186	0.8932	0.8614	97.2022	95.6068	8.5654	8.6178
Hybrid	0.0153	0.0170	0.8900	0.8851	98.6528	97.8772	9.5466	9.6067

Fy								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0155	0.0186	0.9608	0.9496	97.9886	96.4538	6.9486	8.9562
Poly	0.0196	0.0214	0.9375	0.9330	97.1383	96.2369	10.3426	12.3485
Hybrid	0.0173	0.0201	0.9517	0.9409	98.6429	97.9133	11.4941	13.8694

Fz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0495	0.0478	0.4674	0.4794	99.8279	99.7978	26.7682	26.6386
Poly	0.0143	0.0150	0.9555	0.9488	97.1612	97.6135	8.6550	8.7083
Hybrid	0.0134	0.0141	0.9607	0.9549	98.5038	98.7607	9.2570	9.3702

data. The identified ANN F_y model of the indoor MetalBeetle boasts more accurate performance ($R^2 = 0.9496$) and reliable, yet acceptably narrow, PIs ($PICP > 95$, $MPIW = 9.0\%$). Such modelling success is absent for the identified ANN F_z model which sees a poor fit of the measurement data ($R^2 = 0.4794$) with unnecessarily wide PIs ($PICP > 95$, $MPIW = 26.7\%$). Note that the comparatively wider PIs associated with the F_z model over the F_x and F_y counterparts demonstrates the lack-of-confidence with the ANN F_z predictions and demonstrates their utility as a measure of prediction reliability and validity. Nonetheless, the ANN model struggles to reproduce the measured F_z for both the full and test data sets. This ANN F_z model performance is especially disappointing since both the hybrid and polynomial models exhibit significantly better performance metrics.

Indeed, all of the identified polynomial force models of the indoor MetalBeetle perform admirably. The selected model regressors along with their associated coefficients are summarized in tables 15 to 17 in section VIII.D.5 for the normalized models of F_x, F_y and F_z respectively. Also shown in these tables are the additive accuracy gains, in terms of R^2, and stability for each regressor relative to the training data subset. The error residuals and autocorrelation of said residuals demonstrate that the OLS assumptions of zero-mean white noise are satisfied. The error residuals are generally contained within the $1 - \sigma$ bounds as shown in fig. 103. Moreover, the corresponding autocorrelation of error residuals imply white noise in fig. 105.

Despite being the poorest performer, the F_x polynomial model produces a faithful fit of the measurement data ($R^2 = 0.8614$) with reliable PIs ($PICP > 95$, $MPIW = 8.7\%$). The polynomial model of F_y for the indoor MetalBeetle exhibits a better accuracy ($R^2 = 0.9330$) at the cost of wider PIs ($PICP > 95$, $MPIW = 12.4\%$) than its F_x counterpart. Indeed, one expects minimal difference between the F_x and F_y models due to the symmetry and flight characteristics of the quadrotor. However, by the very nature of FPV flight, there is a discrepancy in flight envelopes between the F_x and F_y in the data sets - especially for higher speeds - which influences regressor selection. Such a force asymmetry is apparent through a phase portrait of the force-velocity pairs, shown in fig. 101, constrained the closest 75% of data to the mean. The data is limited to show where the majority of the force measurements lie and is thus more representative of this asymmetry. The identified polynomial F_z model of the indoor MetalBeetle serves as an excellent model, achieving a high accuracy ($R^2 = 0.9488$) with relatively narrow and valid PIs ($PICP > 95$, $MPIW = 8.8\%$).

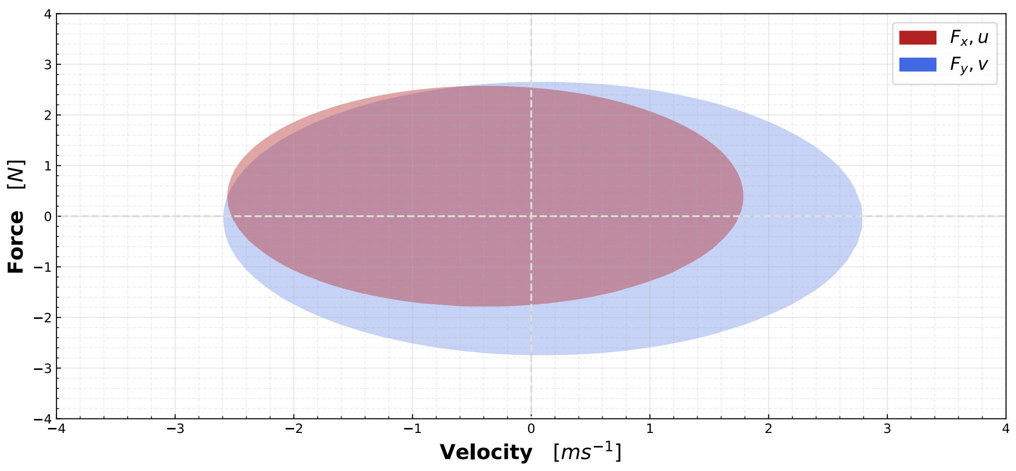


Fig. 101 Phase portrait of the force-velocity pairs, $F_x - u$ and $F_y - v$, of the MetalBeetle during indoor flights demonstrating where 75% of the measurement data, as measured from the mean, is contained.

The identified hybrid force models of the indoor MetalBeetle seek to, and succeed in, improving the accuracies of their underlying polynomial models. The hybrid F_x model provides for a slight boost in model accuracy ($R^2 = 0.8851$) to the levels of its ANN counterpart. In doing so, it also increases the associated PI widths to just shy of 10% and retains their validity ($PICP > 95$, $MPIW = 9.7\%$). Likewise, both the hybrid F_y and F_z models of the indoor MetalBeetle manage improvements over their underlying polynomials with comparative PIs ($R^2 = 0.9409$, $PICP > 95$, $MPIW = 13.9\%$ and $R^2 = 0.9549$, $PICP > 95$, $MPIW = 9.4\%$ respectively). However, the gains of the hybrid models are marginal and stimulate a widening of PIs. Therefore, since the polynomial model performance is already good, perhaps a hybrid approach for indoor force model identification is excessive.

To visualize the performances of the identified indoor MetalBeetle models, fig. 102 depicts the predictive performances of the identified F_x models of the (indoor) MetalBeetle, subject to an untrained FPV flight (Flight 24). Though only the F_x performance is shown here, the other force predictive capacities for this validation flight can be found in the appendix (section VIII.D.5). In accordance with the performance metrics found in table 11, all identified F_x models appear capable of reproducing the measured force of an unseen flight through faithful reproductions of the measured F_x in fig. 102. Indeed, the predictions themselves may not always be in line with the measured forces, however, the measured force is typically contained within the PIs of the models. Nevertheless, there are still some regions wherein the measured force exceeds the PI bounds. Take, for instance, the divergence of the models' predictions and the true force at around $t = 30.5$ s in the highlighted region of fig. 102. From the corresponding video of this flight, the MetalBeetle is seen to be turning sharply in this region. As all models diverge in a similar manner, there is clearly some modelling term that misguides this response or the lack of a term which subsequently corrects for this during the sharp turn. Decomposing the polynomial model of F_x into its cumulative regressor contributions suggests that the control pitching moment, u_q, initially prescribes this behavior. Indeed, it exhibits a high magnitude response at this time in the shape of the subsequent model F_x prediction. Future work should seek the influence, and necessity, of this term on the force models for both LOS and FPV flight.

It is clear from the identified moment model performances in table 12 that the identified force models significantly outperform the identified moment models. Given the poor accuracy metrics, especially for M_y and M_z, the identified models appear to be unable to capture the dynamics behind the aerodynamic moments. However, the PI quality metrics reveal that, even with relatively narrow PIs (indeed, much narrower than the force models), the identified moment models encapsulate the majority of the measured moment dynamics within these PIs. Possible explanations for this discrepancy are that either the moment measurements are heavily contaminated with noise, or that there are not enough excitations, or a combination of the two. The accuracy metrics shown in table 12 do not distinguish between the fit towards useful response and the fit towards noise. Hence, these performance metrics likely predominantly assess the fit with respect to noise.

Therefore, to examine the true model performances, the performance evaluation may be localized to regions of moment excitation. Indeed, the model performance metrics generally improve - especially with respect to the full data

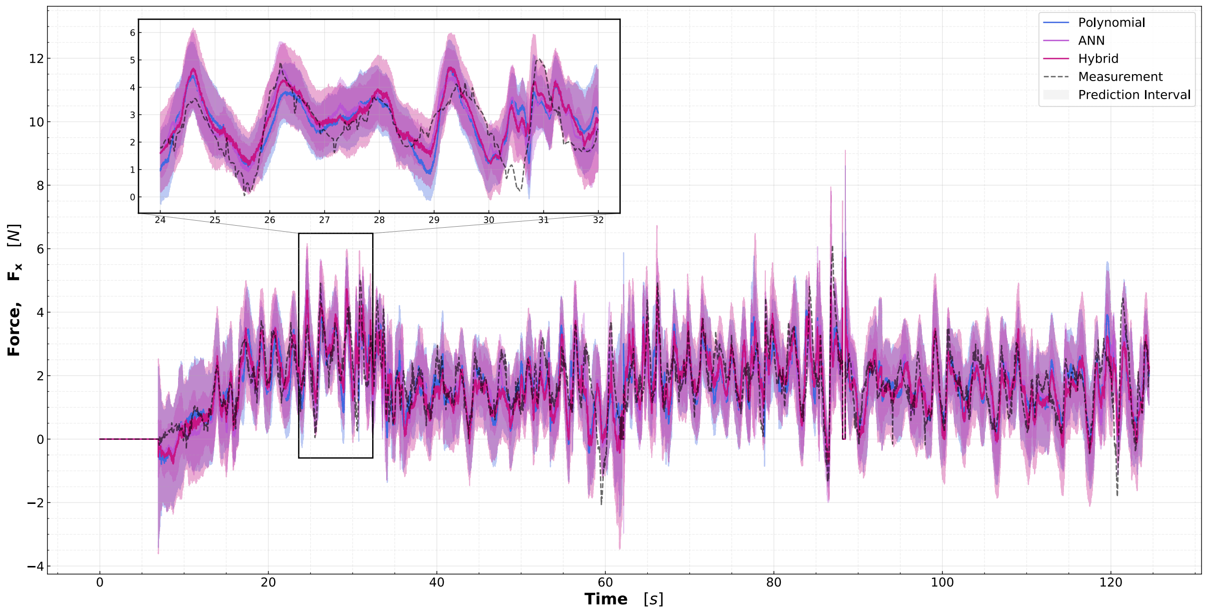


Fig. 102 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_x for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured F_x response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

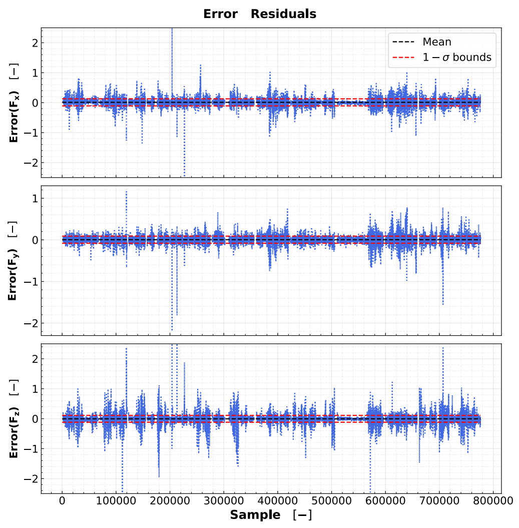


Fig. 103 Error residuals of the identified polynomial force models of the *indoor* MetalBeetle.

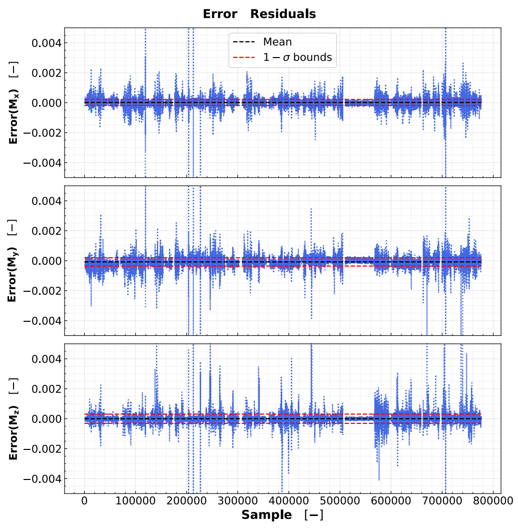


Fig. 104 Error residuals of the identified polynomial moment models of the *indoor* MetalBeetle.

set - around regions of excitation, as shown in table 13. Despite this, the subsequent performances are still poor and there appears to be a significant deterioration in performance from the full to the test data set. This implies that some over-fitting may be occurring or that there are dynamics in the test set which elude the identified models.

Indeed, the ANN models appear to over-fit the training data as they struggle to maintain their accuracy across the full and test data sets while also failing to meet the $PICP \geq 95$ criterion for the test data set. The identified ANN rolling moment, M_x, model offers an objectively lackluster accuracy ($R^2 = 0.3921$) with too narrow PIs ($PICP = 90.4535$,

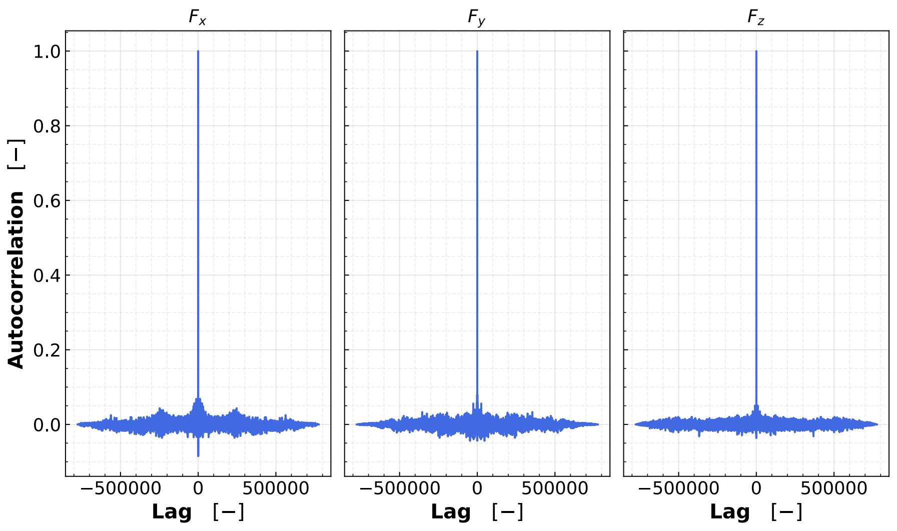


Fig. 105 Autocorrelation of the error residuals of the identified polynomial force models of the *indoor* MetalBeetle.

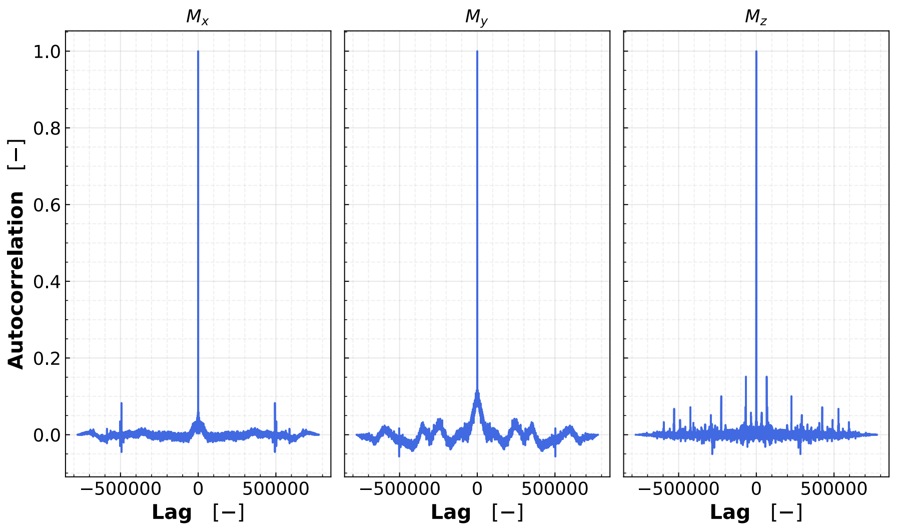


Fig. 106 Autocorrelation of the error residuals of the identified polynomial moment models of the *indoor* MetalBeetle.

$MPIW = 2.8\%$). This model performs much better with respect to the full data set with a decent accuracy and narrow and arguably valid PIs ($R^2 = 0.6737$, $PICP = 94.9 < 95$, $MPIW = 1.6\%$). Interestingly, the MPIW almost doubles for the test set, suitably reflecting the increase in uncertainty for the ANN M_x predictions. However, it cannot be said that the ANN M_x model is valid due to the questionable PI reliability. The deterioration in performance from full to test data sets is even more pronounced for the identified ANN pitching moment, M_y, model with an unacceptable accuracy ($R^2 = 0.0347$) and invalid PIs ($PICP = 81.8 < 95$, $MPIW = 1.2\%$). The ANN yawing moment model also affords a meager model accuracy ($R^2 = 0.1718$) and unreliable PIs ($PICP = 88.4 < 95$, $MPIW = 2.4\%$). The poor $PICP$ performance suggests that, even though the ANN PIs were found to be numerically valid in **Part I: On prediction intervals**, they should still be interpreted with a degree of caution. As such, more research investigating (and ensuring) the validity of the ANN PIs should be conducted in the future. Note, however, that since the PI performance is also poor for the polynomial models the issue may therefore lie in the difficulty faced with identifying moment models in general.

Table 12 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid moment models for the MetalBeetle (Indoor flight). The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Mx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0048	0.0050	0.6540	0.6067	96.9873	96.3316	1.6028	2.4778
Poly	0.0048	0.0051	0.6520	0.5854	99.4874	99.5350	4.3685	6.2730
Hybrid	0.0047	0.0050	0.6605	0.6006	99.7055	99.7518	4.6062	6.6493

My								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0040	0.0049	0.4114	0.0777	94.7818	90.4964	1.2408	1.2821
Poly	0.0054	0.0064	-0.0627	-0.5850	97.9734	97.3046	2.6541	2.6704
Hybrid	0.0049	0.0056	0.1414	-0.2410	99.1271	98.9573	2.9031	2.9269

Mz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0096	0.0098	0.0825	0.0509	91.5682	91.2393	2.0701	2.2006
Poly	0.0164	0.0115	-1.6852	-0.3163	95.9843	96.3398	4.4578	4.4850
Hybrid	0.0163	0.0114	-1.6506	-0.2861	97.4682	97.7894	4.8986	4.9626

The identified polynomial moment models of the indoor MetalBeetle suffer similar performance hardships as their ANN counterparts. The selected regressors, and associated cumulative model R^2 improvements with respect to the training data, are summarized by tables 18 to 20 in the appendix (section VIII.D.5) for the normalized M_x, M_y, and M_z models respectively. Likewise, the error residuals are zero-mean and are generally contained within the $1-\sigma$ bounds as shown in fig. 104. The autocorrelation of said residuals, given in fig. 106, imply minimal correlation among the error residuals. Thus, the OLS assumptions are satisfied.

The performance of the identified polynomial rolling moment, M_x, model closely resembles its ANN equivalent but with valid and wider PIs ($R^2 = 0.3915$, $PICP > 95$, $MPIW = 6.2\%$). In contrast, the identified polynomial pitching moment, M_y, model performs worse than its ANN counterpart with an unacceptable accuracy ($R^2 = -0.1886$) and unreliable PIs ($PICP = 91.5 < 95$, $MPIW = 2.4\%$). Parallels between the polynomial and ANN models continue for the identified yawing moment, M_y, model which suffers poor accuracy and invalid PIs ($R^2 = 0.1580$, $PICP = 93 < 95$, $MPIW = 4.6\%$). The inability of the ANN and polynomial models to produce valid PIs lends support to the hypothesis that much of the issues with the moment models arises from the difficulty in modelling them.

Fortunately, the identified hybrid models are able to improve upon the polynomial model accuracy and extend the PIs to satisfy the $PICP > 95$ condition. This accentuates the instances wherein the adoption of such a hybrid approach is beneficial. For instance, the hybrid rolling moment model improves upon the accuracy ($R^2 = 0.4039$) and PI validity ($PICP > 95$, $MPIW = 6.7\%$) of both the underlying polynomial model. As the hybrid models' performance is heavily intertwined with the polynomial model, the hybrid pitching moment, M_y, model retains a poor accuracy ($R^2 = -0.0463$) but now ensures acceptable PIs ($PICP > 95$, $MPIW = 2.6\%$). The hybrid yawing moment model also sees some performance gains over its underlying polynomial with an improved accuracy ($R^2 = 0.1672$) and valid PIs ($PICP > 95$, $MPIW = 5.1\%$). Overall, the identified moment models clearly struggle to capture the aerodynamic moments, regardless of the employed system identification technique. The conspicuous parallels in poor moment model performance across the different system identification techniques suggests that the issues with identifying moment models lies not with the approximation power of the employed techniques, but rather with the current chosen states or identification data. Indeed, there may be additional states (and interactions therein) missing here which may be integral

Table 13 Summary of the model performances of the identified ANN-only, Polynomial-only, and Hybrid moment models for the MetalBeetle (Indoor flight) *localized around regions of excitation*. The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

Mx								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0080	0.0089	0.6737	0.3921	94.8985	90.4535	1.5525	2.7568
Poly	0.0082	0.0089	0.6598	0.3915	98.7544	98.8469	4.4194	6.1923
Hybrid	0.0078	0.0088	0.6911	0.4039	99.2366	99.3610	4.6952	6.6557

My								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0068	0.0100	0.5149	0.0347	92.1336	81.7736	1.0423	1.1852
Poly	0.0074	0.0111	0.4242	-0.1886	95.2264	91.4842	2.4845	2.3400
Hybrid	0.0068	0.0104	0.5252	-0.0463	97.9163	96.7815	2.7477	2.6094

Mz								
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN	0.0134	0.0181	0.2567	0.1718	92.9966	88.3834	1.6520	2.3490
Poly	0.0124	0.0182	0.3584	0.1580	94.4951	93.0127	4.8706	4.6145
Hybrid	0.0123	0.0181	0.3702	0.1672	96.2987	95.2505	5.1936	5.1406

to describing the moment models. Likewise, the current manoeuvres conducted may be insufficient for adequately exciting the aerodynamic moments. Consequently, future research should investigate potential avenues for improving moment models through augmented state vectors and different input manoeuvres.

Figure 107 depicts the predictive capacities of the identified pitching moment, M_y, models with respect to a test data set (i.e. unseen) FPV flight. Here, the M_y response is shown due to the apparent poor performances of the identified models. Other moment responses for this unseen flight can be found in the appendix (section VIII.D.5). In contrast to poor accuracy observed through table 13, the MetalBeetle models' pitching moment predictions appear to earnestly follow the measured M_y during periods of excitement. This is emphasized by the first highlighted region in fig. 107. However, these moment model predictions appear to be sensitive to noise due to their seemingly arbitrary predictions for periods of minimal, or no, pitching moment excitations. See, for example, the second highlighted region in fig. 107. The poor performance in these regions likely contributes to the underwhelming model performances found in table 13. This may also explain part of the drop in accuracy between the full and test sets, since a larger proportion of the test set consists of noise due to having fewer manoeuvres. Nonetheless, the predictive performance illustrated in fig. 107 shows promise for the usability of the identified moment models. Indeed, the useful moment responses are visibly sparse in fig. 107, thus methods to mitigate the adverse effects of noisy measurements during period of minimal moment excitement should be investigated in future work.

3. Model structures and sensitivities of the indoor MetalBeetle models

In this section, the model structures and sensitivities associated with the identified polynomial, ANN, and hybrid models are presented.

Polynomial model of the indoor MetalBeetle

The identified polynomial F_x models of the MetalBeetle select physically justifiable regressors for the indoor model with more curious choices for the outdoor model. Unsurprisingly, the indoor model also demonstrates more stable

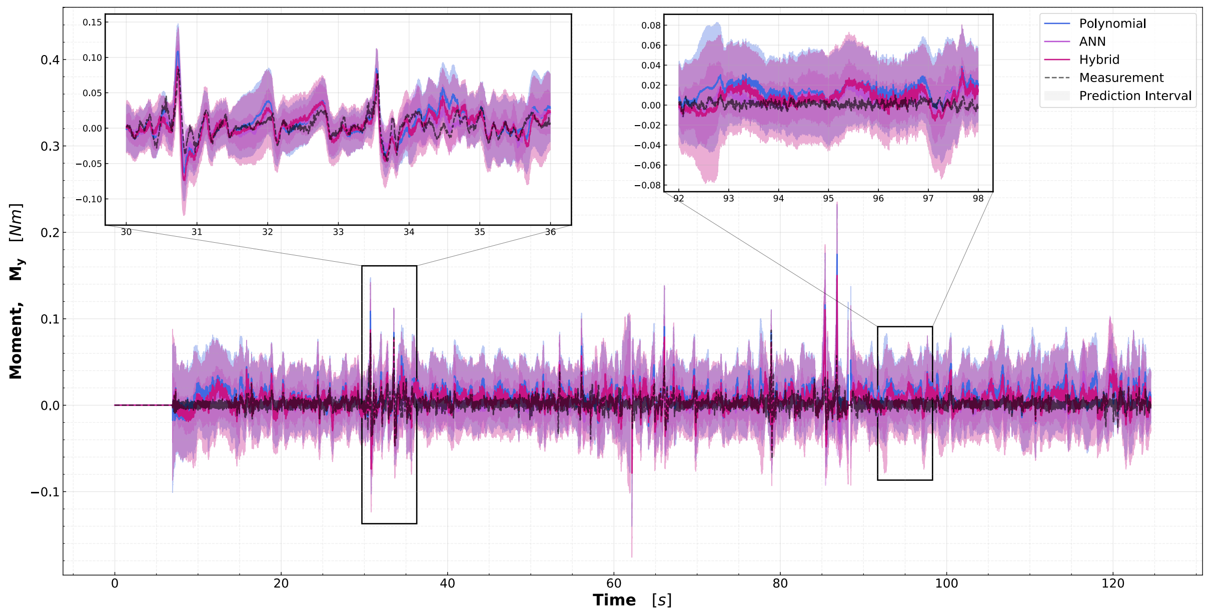


Fig. 107 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_y for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured M_y response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

coefficients. The selected regressors can be found in table 15. The first selected regressor[17] of the indoor model is $\sin(\theta)\mu_{v_{in}}\bar{\omega}_{tot}^3$, which effectively captures dynamics associated forces along x induced by the rotors and interactions therein. It also accounts for much of the modelling accuracy ($R^2 = 0.0099$ to $R^2 = 0.8168$) accentuating the significance of this term and associated effect. The presence of the advance ratio of the induced velocity, $\mu_{v_{in}}$, may also encapsulate some thrust variance and blade flapping effects. Subsequent selected regressors routinely include (combinations of) $\bar{\omega}_{tot}$, μ_x, $\bar{u}$, $\bar{w}$, and θ, all of which are intuitively linked to the underlying dynamics of F_x. Performance improvements are minimal beyond the addition of the sixth regressor, $\cos(\theta)\bar{u}$, and may therefore be removed without detriment to the model. Indeed, the final selected regressor, $\bar{\omega}_{tot}\bar{q}^3$, demonstrates some instabilities due to a relatively high covariance (8.33% of the coefficient's magnitude) and thus could be removed from the model. All other regressors boast low covariances ($< 0.01\%$) which imply model stability.

The indoor polynomial model of the MetalBeetle captures relevant dynamics in a stable model. The associated regressors for the indoor F_y model may be found in table 16. Similar to F_x, much of the indoor F_y model's success is attributable first regressor[18], $\sin(\phi)\bar{\omega}_{tot}^2$, that facilitates motion along y through the thrust vector. This regressor alone improves the accuracy from $R^2 = 0.1128$ to $R^2 = 0.8282$ and is thus a dominant term in explaining the dynamics. Predictably, the rotor speed, $\bar{\omega}_{tot}$ forms a fundamental component of F_y given its pervasiveness in the remaining regressors. Additionally, these regressors incorporate (combinations of) μ_y, p, and $\bar{u}_p$. Note that the reliance of the F_y on $\bar{p}$ and $\bar{u}_p$ likely arises from the side-to-side manoeuvres used to excite the F_y dynamics. Such manoeuvres see a strong correlation between force and associated rotational rates during the transitions in flight direction. Such manoeuvres are more prominent in the F_y data due to the FPV flight. Accuracy gains are minimal following the addition of the sixth regressor, $\bar{\omega}_{tot}\bar{u}_p$, suggesting that the addition of subsequent regressors are likely over-fitting. Nonetheless, all regressors of the F_y polynomial model of the MetalBeetle are accompanied by low covariances $< 1\%$ and thus imply model stability.

The maturity of the analytical models of thrust developed in quadrotor literature is evident through the already decent modelling accuracy ($R^2 = 0.7910$) enjoyed by the fixed regressors of the identified F_z polynomial model of the

[17]Note that this is actually the same first regressor as is selected for the indoor DataCan75 model, which implies that it encompasses some emergent dynamics.

[18]Again, this first regressor is exactly the same as selected for the indoor DataCan75 model, suggesting some emergent dynamics.

indoor MetalBeetle. The selected regressors themselves are summarized in table 17 in section VIII.D.5. The indoor F_z polynomial model of the MetalBeetle successfully builds upon the fixed regressor accuracy through the addition of regressors associated with the total rotor speed, $\bar{\omega}_{tot}$. Most notably, the addition of $\bar{\omega}_{tot}^4$ stimulates a boost in model accuracies from $R^2 = 0.7458$ to $R^2 = 0.9738$. Moreover, the component of thrust that contributes to forces along z while in motion along x and y is captured by the first two selected regressors, $\cos(\theta)\bar{\omega}_{tot}^2$ and $\cos(\phi)\bar{\omega}_{tot}^2$, respectively. Note that these two terms harbor similar coefficient values which arise from the symmetry of the quadrotor. In theory, these coefficients should be identical, but are likely different here due over-fitting. The prevalence of the higher order terms associated with $\bar{\omega}_{tot}$ implies that the rotor speed, intuitively, constitute much of the underlying dynamics. Other ubiquitous states of note are the control moments, $|\bar{u}_p|$, $|\bar{u}_q|$, and $|\bar{u}_r|$, and interactions therein which evidently capture some underlying phenomena. While it is unclear exactly what dynamics these terms encapsulate, it is suspected to relate to some variance in thrust induced by the controlled rotations of the quadrotor. Indeed, such rotations directly result from thrust differentials and involve interactions between rotor wakes. All of the selected regressors for the indoor MetalBeetle polynomial model of F_z are accompanied by low covariances ($< 0.2\%$) which indicates model stability.

The indoor MetalBeetle polynomial model of M_x selects reasonable regressors that are stable. The selected regressors themselves may be found in table 18. Intuitively, the control rolling moment, $\bar{u}_p$, total rotor speed, $\bar{\omega}_{tot}$, form components of many of the selected regressors. Indeed, the indoro model shares some common regressors with the equivalent outdoor model, such as $\bar{\omega}_{tot}\bar{u}_p$, $\bar{\omega}_{tot}\bar{p}$, and $\bar{\omega}_{tot}\bar{u}_p|\bar{u}_r|$, which advocates for their necessity in describing the underlying dynamics. This is especially likely for $\bar{\omega}_{tot}\bar{u}_p$ since both the indoor and outdoor models select it first. Moreover, low covariances (i.e. $< 0.01\%$ of the associated coefficient's magnitude) imply that the polynomial model of M_x is stable.

In contrast, the pitching moment, M_y, polynomial model of the indoor MetalBeetle harbors regressors with instabilities. These regressors may be found in table 19. Analogous to the M_x models, many of the selected regressors are constructed with the control pitching moment, $\bar{u}_q$, and the total rotor speed $\bar{\omega}_{tot}$. The indoor model selects $\bar{\omega}_{tot}\bar{u}_q$ as the first regressor, paralleling the choice of the equivalent outdoor model and accentuating its importance in capturing the underlying dynamics. Recall that $\bar{\omega}_{tot}\bar{u}_p$ was also selected first for the M_x model. This strongly suggests that the linear interaction between $\bar{\omega}_{tot}$ and the corresponding control moment is descriptive of some of the moment dynamics both indoors and outdoors. Peculiarly, the indoor model selects $\bar{\omega}_{tot}|\bar{p}|^3\bar{q}$ as a regressor, which is driven primarily by the roll rate, $\bar{p}$, and should have a negligible effect on the pitching moment in this capacity. Indeed, this regressor holds the highest covariance (3.4% of its associated magnitude) in the M_y model only amplifies its debatable addition. Hence, the reliability of the identified indoor polynomial model of M_y should be interpreted with caution.

The identified yawing moment model of the indoor MetalBeetle, M_z, selects both sensible and questionable regressors culminating in some instabilities. The indoor yawing moment model regressors can be found in table 20. This model intuitively selects many terms composed, in part, of the control yawing moment, $\bar{u}_r$, the yaw rate, $\bar{r}$, and the total rotor speed, $\bar{\omega}_{tot}$. Interestingly, the previously observed conspicuous linear interaction of $\bar{\omega}_{tot}\bar{u}_r$ is replaced by a higher order variant, $\bar{\omega}_{tot}\bar{u}_r^3$, and cast doubt on the importance of the linear relation of these two states but nonetheless supports an emergent effect founded in their interaction. Unexpectedly, the yaw angle debuts in the indoor model through the terms $\cos(\psi)\bar{u}_r$ and $\sin(\psi)\bar{u}_r$. However, in this case, such a dependence is probably a consequence of over-fitting given the similar coefficient magnitudes and inclusion of both sin and cos. Hence, the model likely attempts to capture effects associated with $\bar{u}_r$ but modulates it, coincidentally, with the yaw angle for a better fit. However, the indoor yawing moment model may be considered unstable given the high covariance of $\bar{\omega}_{tot}|\bar{p}|^3|\bar{q}|$ (4.2% of the associated coefficients magnitude).

Hybrid model of the indoor MetalBeetle

The weight magnitudes of the input layer, averaged over the last 10% of training epochs (i.e. 15 epochs), provides insight into any emergent effects recognized by the ANN compensators while the variance therein gives an indication of stability. These weight magnitudes and variances are respectively illustrated in fig. 108 and fig. 109 for the hybrid force models. Likewise, the input weight magnitudes of the indoor moment models are illustrated in fig. 110 with corresponding weight variances in fig. 111.

The marginal improvements of the identified indoor hybrid model of F_x are reflected by almost uniform weights with adequate variances. Although similar in magnitude, many of the higher weight inputs of F_x in fig. 108 correlate well with terms expected to be relevant for describing this force. Hence, the ANN compensator may still be extracting useful dynamics from the polynomial error residuals. For example, the hybrid model allocates some importance to the quadrotor's velocities of $\bar{u}$ and $\bar{w}$, which see minimal use in the polynomial model. The cohesive magnitudes of these inputs suggests that no one term is dominant and implies that the ANN compensator capitalizes on interaction effects

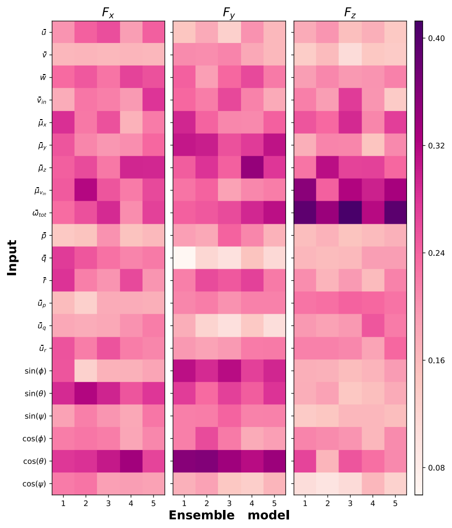


Fig. 108 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the ANN compensator component of the identified hybrid force models of the MetalBeetle (*Indoors*).

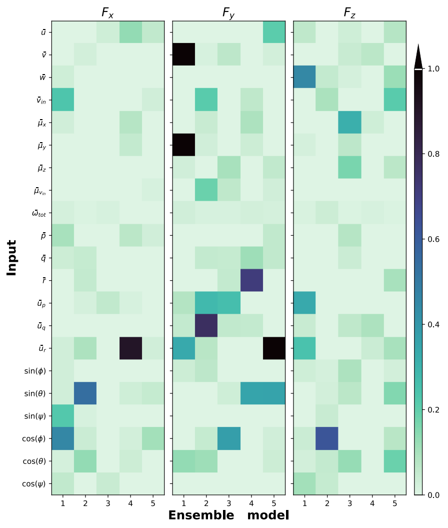


Fig. 109 Variance in input layer weights, taken over the last 10% of training epochs, of the ANN compenstor component of the identified hybrid force models of the MetalBeetle (*Indoors*).

between these inputs. Moreover, both trigonometric identities of the pitch angle hold comparatively high magnitudes, which lends support to the recognition of interaction effects (e.g. through interactions of $\sin(\theta)\bar{\omega}_{tot}$). However, these uniform weights may also be indicative of over-fitting and the inability of the ANN compensator to identify any emergent dynamics in the polynomial error residuals. While the generally low variances of the weights implies convergence and stability in the model, there are some high variance terms, particularly for the attitude inputs, which are a cause for concern and question the stability of the model.

Similarly, the reliability of the ANN compensator of the indoor F_y model of the MetalBeetle is dubious, given the allocation of high weight magnitudes to questionable inputs and general weight instabilities. Despite this, the hybrid model also reasonably associates high weight magnitudes with the y- and z-component of the advance ratio, μ_y and μ_z respectively, and the roll angle through $\sin(\phi)$. In the context of the improved accuraciy with respect to the underlying polynomial for both the full and test data sets, it is possible that the ANN compensator identifies some useful dynamics - likely associated with the aforementioned inputs - from the polynomial error residuals. However, peculiarly, the ANN compensator consistently assigns the highest weight magnitudes to $\cos(\theta)$. This is probably consequence of over-fitting, given that the pitch angle should have a negligible influence on forces along this axis. Additionally, high variances associated with many of the input weights suggest that the model has yet to converge or is unstable. Both scenarios are undesirable as they may lead to erroneous predictions. Therefore, despite the improvements gained by the ANN compensator of the indoor MetalBeetle hybrid model of F_y, unexpected weight magnitudes and poor stability characteristics challenge the reliability of the model.

In contrast to the predominately uniform weight magnitudes of the indoor MetalBeetle hybrid models of F_x and F_y, the corresponding F_z hybrid model demonstrates clear input importance. The associated weight variances also support model stability. The input magnitudes of the ANN compensator show a strong dependence on $\bar{\omega}_{tot}$, $\mu_{v_{in}}$ and μ_z. These inputs almost coincide exactly with the fundamental terms of the analytical models of thrust variance and blade flapping. Therefore, the ANN compensator of F_z may be targeting these phenomena. Even if this is not the case, the utility of these inputs towards F_z is obvious. To a lesser extent, the ANN compensator also allocates some importance to μ_x, which is reflective of the FPV flight and may also be relevant for modelling blade flapping. Through this, the hybrid F_z model is able to improve upon the performance of its underlying polynomial model. However, there are a number of high variance weights in the hybrid model which may reflective of over-fitting and could induce model instability.

The identified hybrid rolling moment, M_x, model of the indoor MetalBeetle associates consistent importance to certain inputs with generally lower variances than the force models. Intuitively, the indoor hybrid model recognizes the significance of the rolling control moment, $\bar{u}_p$, followed by the roll rate, $\bar{p}$, and total rotor speed, $\bar{\omega}_{tot}$. Although the

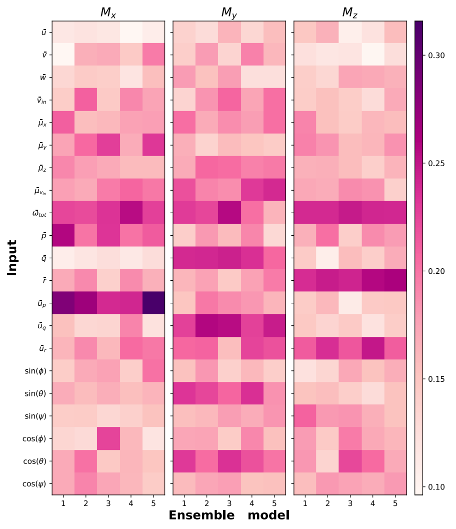


Fig. 110 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the ANN compensator component of the identified hybrid moment models of the MetalBeetle (*Indoors*).

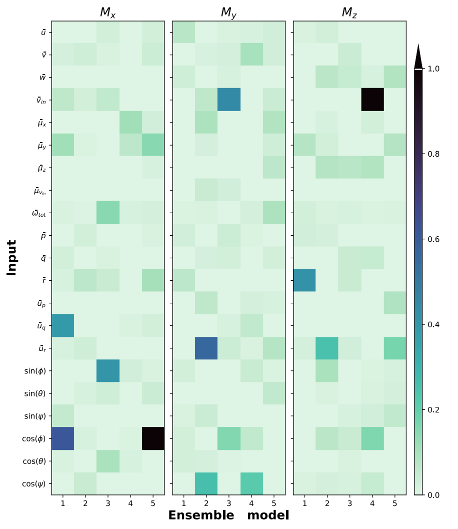


Fig. 111 Variance in input layer weights, taken over the last 10% of training epochs, of the ANN compenstor component of the identified hybrid moment models of the MetalBeetle (*Indoors*).

associated weight variances are much lower than those of the force models, there are a few high variance inputs (i.e. $\approx 80\%$ of the associated weight magnitude) concentrated around the attitude inputs, which are a cause for concern.

The identified indoor hybrid pitching moment, M_y, model exhibits arguably better model stability with a sensible selection of important inputs. For instance, the model selects $\bar{u}_q$, $\bar{q}$, and the pitch angle, θ, which relate well to the underlying physical phenomena. Other inputs, such as $\bar{\omega}_{tot}$, also afford high weight magnitudes among many of the constituent ANNs. The indoor hybrid model of M_y also harbors mostly low weight variances which implies stability. However, some high variances, particularly those associated with the yaw rate, $\bar{r}$, are concerning given the apparent importance of this input afforded by the M_y model. Therefore, the subsequent predictions may be unreliable.

Likewise, the indoor hybrid yawing moment, M_z, model of the MetalBeetle exhibits generally low weight variances with some concerning high variance inputs. Furthermore, defined weight magnitudes demonstrate clear input significance. The indoor model recognizes the relevance of the total rotor speed, $\bar{\omega}_{tot}$, the control yawing moment $\bar{u}_r$, and the yaw rate, $\bar{r}$. All of these inputs understandably have some influence on the yawing moment. The indoor hybrid M_z model harbors a single concerning high variance term, which although associated with a low weight magnitude term, may nonetheless propagate instabilities throughout the rest of the model given the dense ANN architecture.

ANN model of the indoor MetalBeetle

The weight magnitudes, averaged over the last 10% of training epochs, and the variances therein are illustrated in fig. 112 and fig. 113 respectively for the identified indoor MetalBeetle ANN force models. Figure 114 and fig. 115 respectively depict the input weight magnitudes and corresponding variances for the identified ANN indoor moment models of the MetalBeetle.

The identified indoor MetalBeetle ANN F_x model allocates the highest weight magnitudes to the total rotor speed, $\bar{\omega}_{tot}$, advance ratio of the induced velocity, $\mu_{v_{in}}$, and the pitch angle through $\sin(\theta)$. These inputs, and interactions therein, presumably relate to effects of the rotor system on the forces along the body x-axis as modulated by the pitch angle. This is in agreement with the equivalent polynomial models for which these states form integral regressors. While the input weight variances are mostly low for the indoor model, the consistency of the variances among the ensemble models for $\sin(\theta)$ points to some instabilities with respect to this input, which is undesirable. As this is an integral modelling term, conclusions of stability for the ANN indoor F_x model should be interpreted with caution.

Likewise, the identified indoor MetalBeetle ANN F_y model recognizes clear input importance but suffers some stability issues. As expected, the indoor ANN model of F_y assigns high weights to inputs such as $\bar{\omega}_{tot}$, $\mu_{v_{in}}$, μ_y, and the roll angle, through $\sin(\phi)$. Such selections are reasonable and well-founded in physical phenomena. Again, these inputs

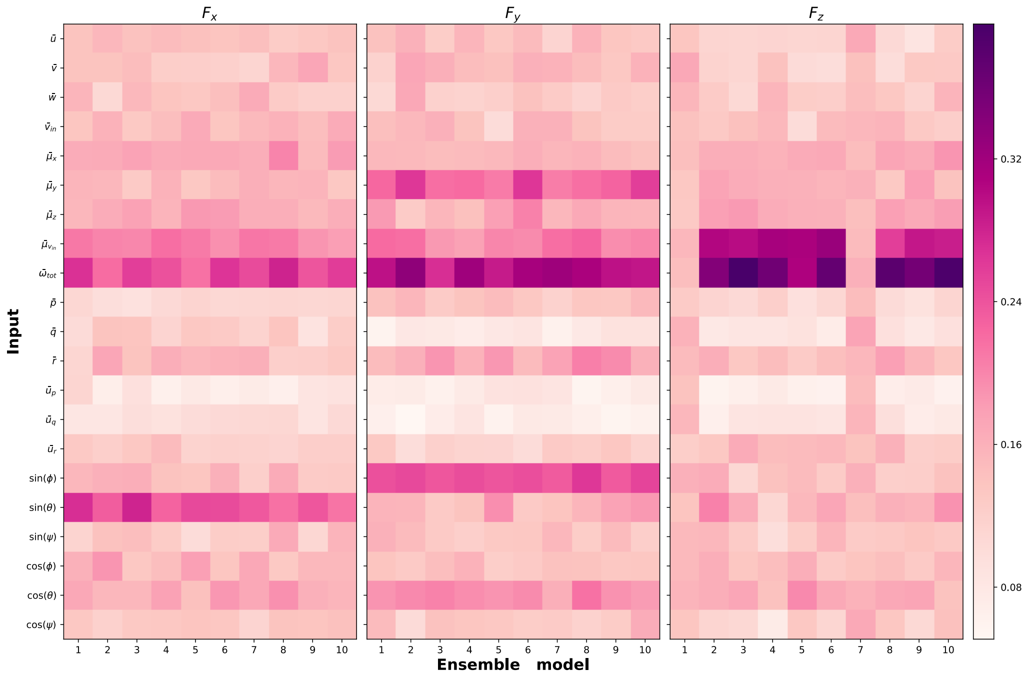


Fig. 112 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the identified ANN force models of the MetalBeetle (*Indoors*).

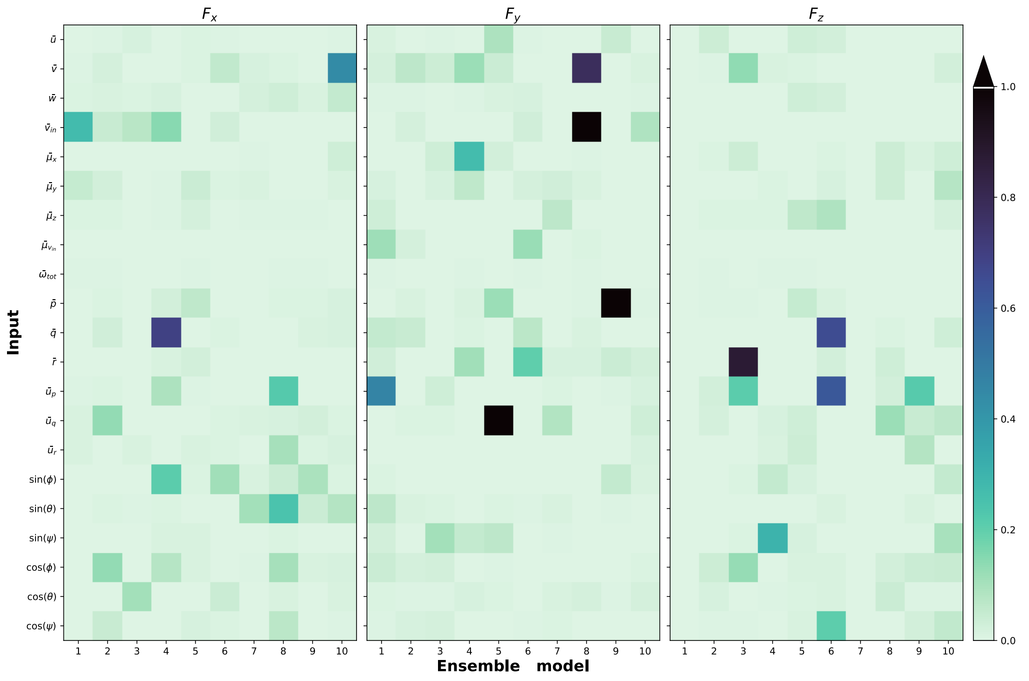


Fig. 113 Variance in input layer weights, taken over the last 10% of training epochs, of the identified ANN force models of the MetalBeetle (*Indoors*).

Table 14 Original and Modified (i.e. through the removal of the unconverged ANNs) F_z ANN model performances for the MetalBeetle (Indoor flight). The NRMSE and R^2 describe the accuracy of the model predictions with respect to the full and test data sets. Also shown are the quality metrics for the model prediction intervals in the form of the PICP and MPIW.

	Fz							
Model	**NRMSE**		**R^2**		**PICP**		**MPIW**	
	Full	*Test*	*Full*	*Test*	*Full*	*Test*	*Full*	*Test*
ANN (Original)	0.0495	0.0478	0.4674	0.4794	99.8279	99.7978	26.7682	26.6386
ANN (Modified)	0.0131	0.0149	0.9625	0.9494	98.1166	97.6724	5.0070	5.5589

draw strong parallels to the indoor polynomial model counterpart. However, the indoor F_y ANN curiously affords some modelling significance to the pitch angle, through $\cos(\theta)$, and may indicate over-fitting. Indeed, the indoor F_y model harbors some strong instabilities (variance of $> 100\%$ of the associated weights magnitude) for a few of its neurons which is a cause for concern and casts doubt on its reliability.

What is immediately apparent about the indoor ANN F_z model is that two of the constituent ANNs have not converged to a solution, leading to an invalid model. This is evident through their lack of distinctive weight magnitudes with low associated variances. In fact, if these poor ANN models are removed, then the performance of the indoor ANN F_z model of the MetalBeetle improves drastically to the levels of its polynomial and hybrid counterparts, as shown in table 14. With these ANNs removed, the indoor ANN F_z model rather intuitively allocates the greatest input importance to $\bar{\omega}_{tot}$ and $\mu_{v_{in}}$. Most other inputs appear in batches of importance. Again, these inputs are reminiscent of the analytical descriptions of quadrotor thrust models and the selected regressors of the underlying polynomial. Consistent with the other ANN indoor force models, the F_z model harbors some high variance inputs which imply model instability. Moreover, note that the removal of the unconverged ANNs also amplifies the effects of these instabilities which deteriorate its reliability.

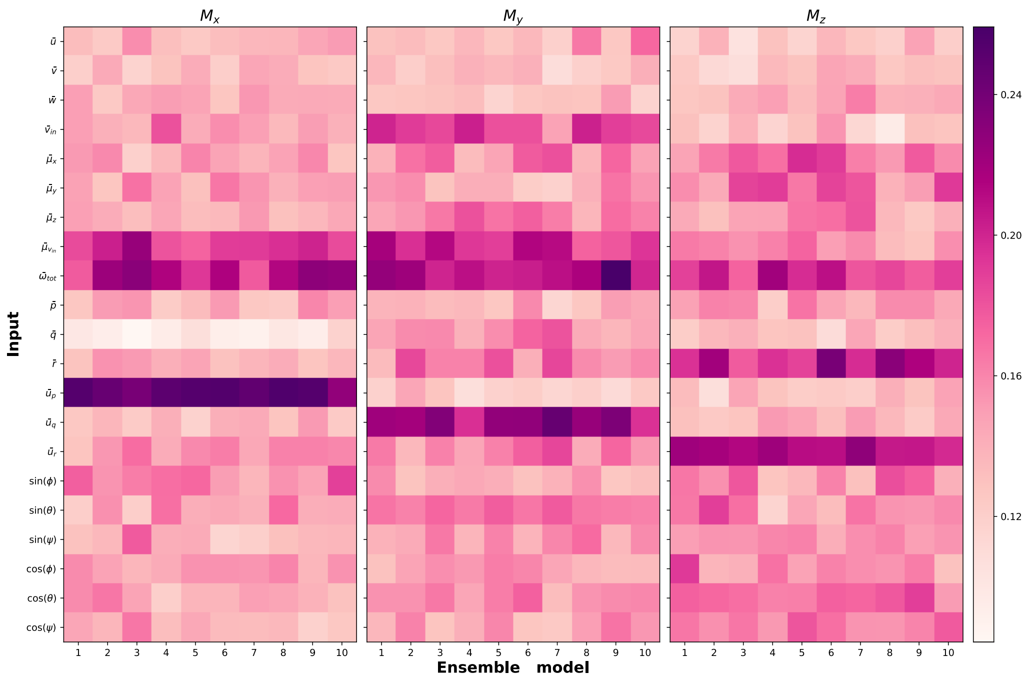


Fig. 114 Input layer weight magnitudes, averaged over the last 10% of training epochs, of the identified ANN moment models of the MetalBeetle (*Indoors*).

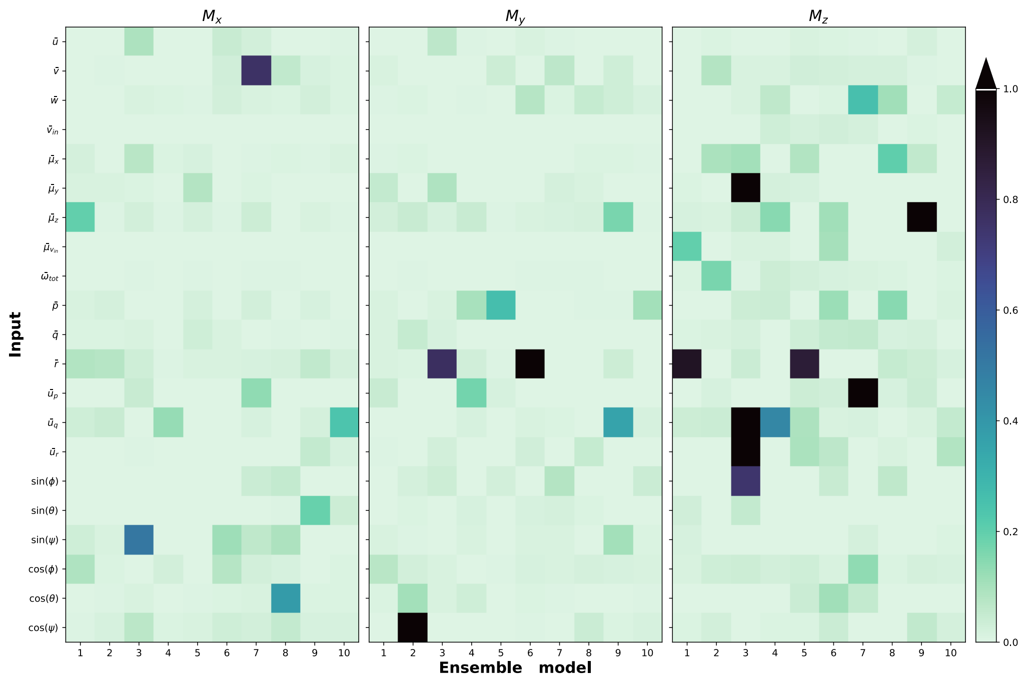


Fig. 115 Variance in input layer weights, taken over the last 10% of training epochs, of the identified ANN moment models of the MetalBeetle (*Indoors*).

The identified indoor ANN rolling moment, M_x, model of the MetalBeetle draws inspiration from the fundamental physical phenomena with scattered high variance weights. The indoor ANN M_x model of the MetalBeetle reasonably elects the control rolling moment, $\bar{u}_p$, the total rotor speed, $\bar{\omega}_{tot}$, and the advance ratio of the induced velocity, $\mu_{v_{in}}$, as the most significant inputs. Indeed, strong parallels to the polynomial and hybrid models are apparent through the selection of $\bar{u}_p$ and $\bar{\omega}_{tot}$, however, the choice of $\mu_{v_{in}}$ is more peculiar. It may be the case that there are some interaction effects associated with $\mu_{v_{in}}$ that are absent in the candidate regressors of the polynomial models or similarly captured by different regressors. Nonetheless, this apparent importance of $\mu_{v_{in}}$ should be investigated in further research as it may relate to blade flapping, or even as of yet un-described, rotor interaction effects. While the sparse and scattered high variance weights are not associated the significant inputs, they may nonetheless induce some instabilities in the M_x model.

Similarly, the identified ANN indoor M_y, model of the MetalBeetle shows a reliance on expected inputs but exhibits more instabilities than its M_x counterpart. The control pitching moment, $\bar{u}_q$, emerges as the most important input followed by $\bar{\omega}_{tot}$, $\mu_{v_{in}}$, and $\bar{v}_{in}$. Hence, these may be considered integral modelling terms in a manner reminiscent of its M_x counterpart. The extended reliance on $\bar{v}_{in}$ may stem from the lower speed flight along the quadrotor's y-axis, for which $\mu_{v_{in}}$ is less prominent, but nonetheless address similar dynamics. It should be noted that one of the constituent ANNs of the indoor model suffers a relatively high variance associated with $\bar{u}_q$, which is undesirable and may induce some unexpected variability in moment predictions. Subsequently, the indoor M_y ANN model may be argued to be invalid.

The distributed weight magnitudes, and high associated variances, of the identified indoor ANN yawing moment, M_z, model suggest that it struggles to capture the underlying dynamics. Nonetheless, some emergent inputs are still visible. For example, the importance of the yawing control moment, $\bar{u}_r$, the total rotor speed, $\bar{\omega}_{tot}$, the yaw rate, $\bar{r}$, are recognized. These inputs correlate well with physical phenomena. While the indoor ANN M_z also harbors many low variance weights, those with high variances are critical for the model predictions. Take, for example, the extremely high variances ($\approx$ 100% of the corresponding weights magnitude) associated with $\bar{r}$ and $\bar{u}_r$. This is unacceptable and implies an invalid indoor ANN M_z model.

4. Simulation of indoor MetalBeetle models

Each of the identified models of the indoor MetalBeetle are first subject to a hovering simulation to evaluate their responses and tune controllers. Subsequently, simulations are run to determine the models' responses to oscillating attitude step input commands. The subsequent simulation results of the identified polynomial-only, hybrid, and ANN-only indoor models of the MetalBeetle are summarized here.

Note that, despite the influences of wind, and differences between the indoor and outdoor polynomial model structures of the MetalBeetle, almost the same PID controller is found to be suitable for controlling both models. Slight modifications were made to give a better response in terms of damping-out oscillations. Nonetheless, the transferability of the controller implies that the identified models are similar and reflects expectations for the true quadrotors. This also accentuates the utility of the system identification pipeline.

Polynomial model

The commanded rotor rate and attitude response of the identified polynomial model of the indoor MetalBeetle for simulated hovering flight are illustrated in fig. 116 and fig. 117 respectively. As the rotor speeds are initialized around their hovering values, they are expected to remain relatively constant. This is indeed the case. Moreover, the attitudes tend to zero. While there is an obvious initial response in the attitudes, this arises from the quadrotor stabilizing itself due to its moment of inertia asymmetries. As such, the attitudes and rotor speeds are reflective of reality for hovering flight.

The corresponding force and moment responses are depicted in fig. 118 and fig. 119 respectively. Note that, the confidence intervals are large here since hovering flight is technically outside the envelope of the identification data [19]. Instabilities are apparent in the force response, where there are some runaway terms in the polynomial model for F_x. This is due to the snowballing effect of velocity terms in the model which increase alongside the force. Therefore, an increase in force inflates the resultant velocity which increases the force further, and so on. To illustrate this effect, refer to the regressor highlighted in black in fig. 121, which depicts the cumulative contributions of the F_x polynomial regressors - in descending order of the legend. Note that, due to the corrections made by the controller to maintain attitude, the F_x tends negative during hovering flight. However, when given even a slight positive velocity, it instead increases positive. Upon further analysis of the roots of this instability, it was found that removing these velocity terms altogether did not eliminate the behaviour as other model regressors simply replaced the runaway phenomena. Through this, it is likely that the issue lies with the nature of the F_x-based flight. As such, F_x polynomial indoor models were identified on LOS-only and FPV-only flight, but both models still exhibited this runaway behaviour. It is therefore

[19]Recall, that only excitations were used for identification which inherently excludes hovering flight.

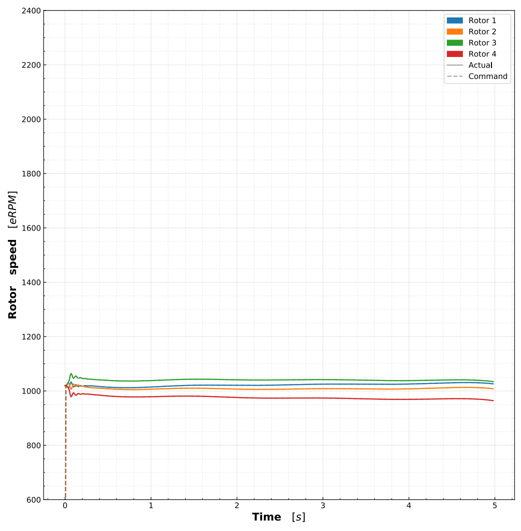


Fig. 116 Commanded rotor speed of the identified indoor polynomial model of the MetalBeetle during hovering flight.

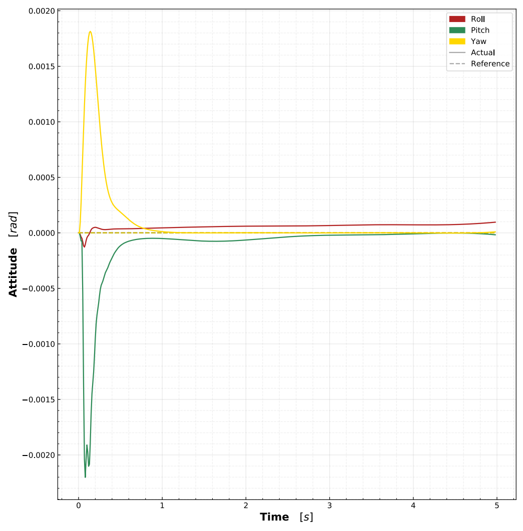


Fig. 117 Attitude response of the identified indoor polynomial model of the MetalBeetle during hovering flight.

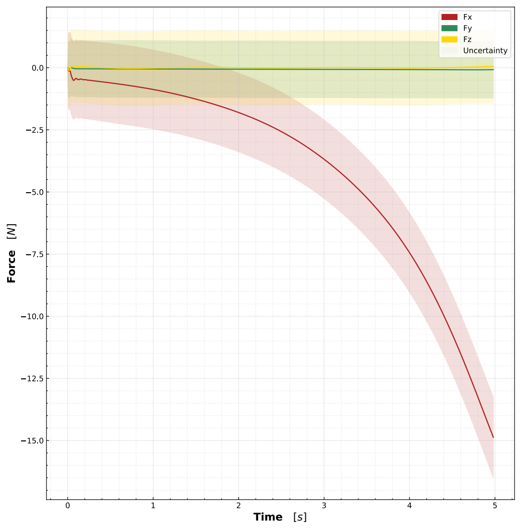


Fig. 118 Force response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle during hovering flight.

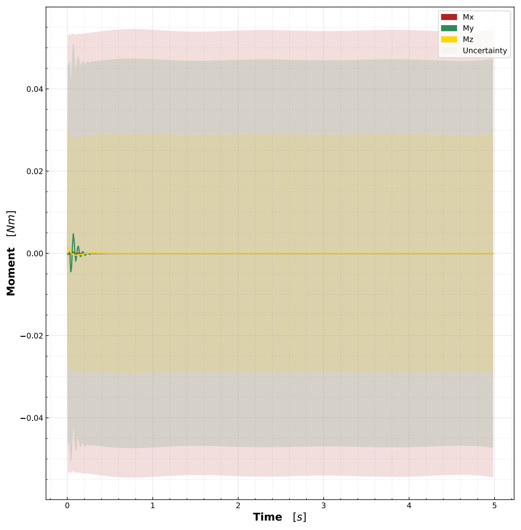


Fig. 119 Moment response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle during hovering flight.

tempting to propose that this issue is related to the lack of drag-based terms in the model, especially given the low speeds of flight. While this may indeed be a contributing factor, the F_y does not suffer such instabilities and is subject to similar manoeuvres. Consequently, the F_y model structure was also applied to the F_x model, yet the runaway problems persisted. Indeed, the issue lies with the F_x data itself and may be associated with the force asymmetries of F_x towards the positive forces in the phase portrait of the force-velocity pairs of $F_x - u$ and $F_y - v$ in fig. 120. Here, the different contour lines denote where 95%, 75% and 50% of the data is contained with decreasing opacity respectively. The exact reasons for this asymmetry are unknown but the inherently forward-based nature of FPV flight likely amplifies this asymmetry. Moreover, it was also noted that the gyroscope biases would drift during flight following system identification manoeuvres. This drift may also influence this asymmetry. In contrast, the moment responses appear sensible and remain near zero after initial attitude corrections.

The true test of the models' capabilities is evaluated through their responses to oscillating attitude step inputs. While simplistic, these manoeuvres are similar to those conducted for data collection (refer to pitch commands in fig. 129) and, thus, the models should be valid for such flight. As the roll and pitch responses are largely the same, the identified polynomial model of the indoor MetalBeetle is subjected to combined roll and pitch step commands.

The commanded rotor speeds and resultant attitude of the MetalBeetle are depicted in fig. 122 and fig. 123 respectively. The attitude response reveals that the model is tracking the step references appropriately and that the controller is therefore working as intended. Both the pitch and roll angles follow the references in a similar fashion and in a manner that reflects the quadrotor's true response to such step inputs. Slight differences are expected and arise from asymmetries in the moment of inertia of the quadrotor. The seemingly arbitrary step attitude inputs are designed to restrict the quadrotor to the velocity limits of the identification data such that the models remain valid. Moreover, the commanded rotor speeds also respond as expected whereby sharp changes in rotational rates occur to initiate an attitude change. The average rotor speed also increases during periods of non-zero roll and/or pitch to compensate for gravity, as expected. Note that the rapid oscillations of the rotor speeds during the step inputs are due to the sub-optimal PID gains. Moreover, these (linear) PIDs were tuned around hover and may therefore not be suitable to gracefully control the non-linear quadrotor when turning.

Nevertheless, the quadrotor appears to respond to the step inputs as intended and the resultant force response is illustrated in fig. 124. Given that the BetaFlight axis system is defined as x-forward, y-left, and z-up, the simulated force responses are reasonable and reflect what may be expected following a step response. As the quadrotor rolls and pitches, the F_z increases as necessary to maintain altitude. Likewise, a positive roll angle stimulates a negative F_y while a positive pitch angle results in a positive F_x. The nature and shape of these force responses is also reasonable as they mirror those found in the identification data (refer to fig. 129). What is perhaps peculiar is the rapid oscillatory

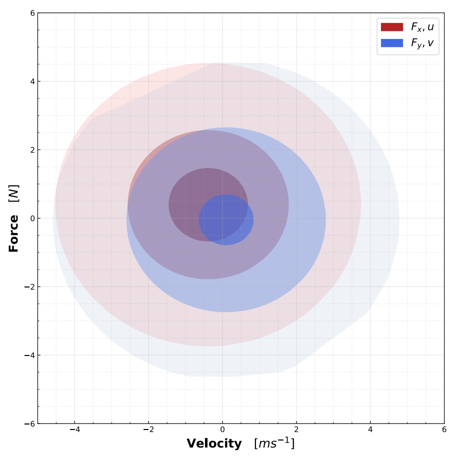


Fig. 120 Phase portrait of the force response versus velocity of F_x,u (in red) and F_y,v (in blue) pairs for the indoor MetalBeetle identification data. The proportions of data contained within the contour bounds are encoded through decreasing opacity for 95%, 75%, and 50% of the data respectively.

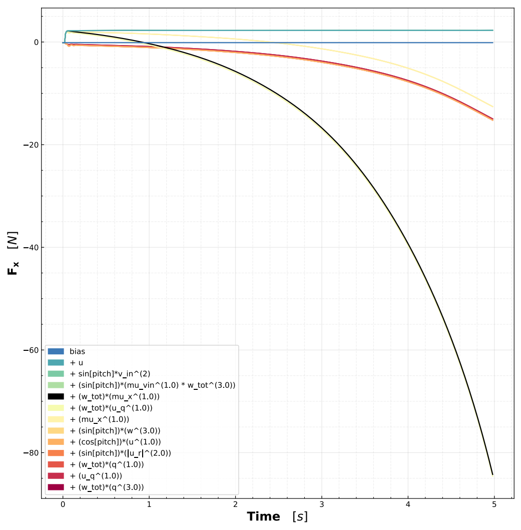


Fig. 121 Cumulative contributions of the constituent regressors of the identified polynomial model of F_x for the indoor MetalBeetle during simulated hovering flight. Highlighted in black is the runaway regressor which induces model instabilities. Regressor contributions are in descending order in the legend.

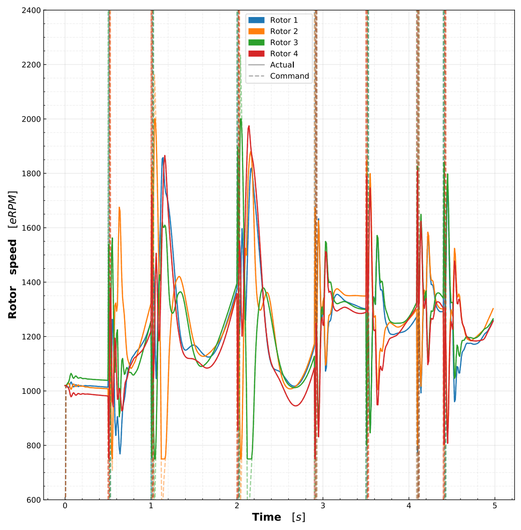


Fig. 122 Commanded rotor speed of the identified indoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

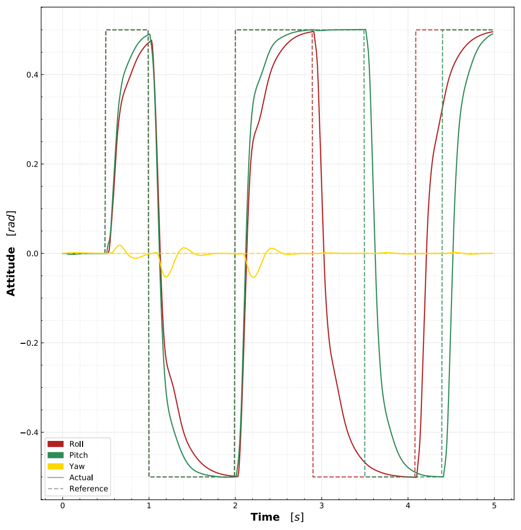


Fig. 123 Attitude response of the identified indoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

behaviour of the force responses during the step transitions. This behaviour is due to poor tuning of the controllers for such manoeuvres, which induces rapid oscillations in the commanded inputs as shown in fig. 126. These oscillations are introduced to the force models primarily through the rotational rate and control moment terms. For instance, consider the behaviour of the regressor $\bar{\omega}_{tot}\bar{u}_q$ of the F_x model in fig. 127. Figure 127 illustrates the cumulative contributions of the regressors following the descending order of the legend with the oscillatory regressor highlighted in black. After a commanded step transition, the force responses progress smoothly as the quadrotor accelerates in the appropriate

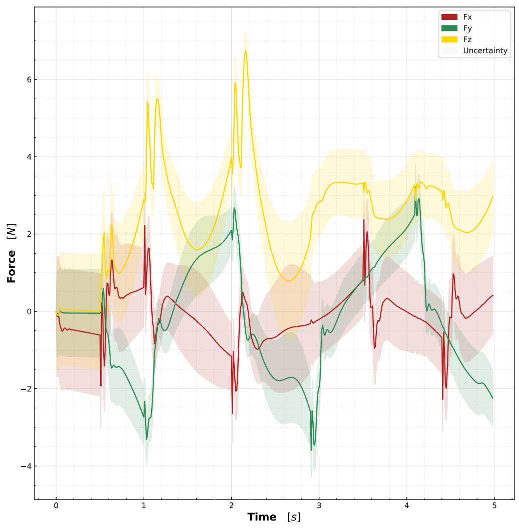


Fig. 124 Force response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

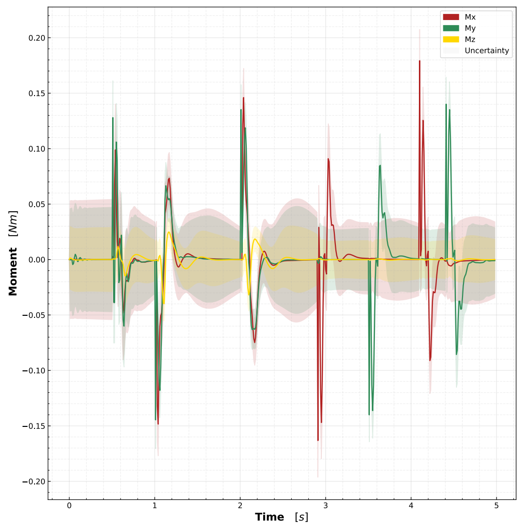


Fig. 125 Moment response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

direction.

The simulated moment model responses are shown in fig. 125 and appear to contradict the poor performances seen in section VIII.D.2. The identified polynomial moment models produce the expected sharp sine-wave like pulses and subsequent corrections when subject to step inputs and resemble those found in the measurement data, as shown in fig. 134. It is noteworthy to highlight the utility of the PIs here. They show a caterpillar-like shape whereby they increase during periods of no moment excitations - indicating low confidence - and decrease during periods of moment excitation, exhibiting high confidence. This is reflective of the training data wherein these models were identified through such excitations.

Indeed, it is difficult to visualize what these forces and moments entail for the quadrotor. Consequently, fig. 128 depicts the resultant velocity response. In accordance with reality, the velocity changes direction and accelerates smoothly along with the corresponding transitions in attitude. The differences between the F_x and F_y models are apparent through these velocity responses whereby the velocity along x, u, is more damped than that along y, v. This is likely a consequence of the runaway F_x term making it more difficult for u to change direction and may encompass drag or rotor wake effects. The velocity responses in particular show a strong resemblance to the measured velocity along x, u, in the identification data when subject to similar pitch inputs as depicted in fig. 129. Indeed, the shape of the velocity responses are similar around the regions of step transition. To a lesser extent, the sharp changes in velocity towards zero following a direction change are also evident in both the simulation responses and the measurement data. These obvious parallels imply that the identified models capture the dynamics well. Note, however, that in the identification data, the thrust was also increased following a direction change to expedite this transition and accelerate the quadrotor. This may explain some of the discrepancies between the simulation and reality, especially in terms of the higher magnitudes associated with the identification data and rapid initial changes in velocity following direction changes. Nonetheless, the models perform faithfully and, despite the runaway F_x, appear to reflect reality well in periods of motion.

To complete the attitude response of the identified polynomial models of the indoor MetalBeetle, a oscillating step input to the yaw angle is also simulated. The corresponding commanded rotor speeds and resultant attitude are illustrated in fig. 130 and fig. 131 respectively. Again, it is clear from the attitude response that the identified polynomial model is capable of realistically tracking the commanded yaw angles. Peculiarly, the average rotor speeds decrease over time when they are expected to remain somewhat constant. However, the yawing manoeuvre induces an upwards velocity which the altitude controller subsequently tries to control. A net positive thrust is generated since the rotor speed lower saturation limit is closer to hover than the upper bound. Towards the end of the rotor response, the rotor speeds return to idle thrust levels.

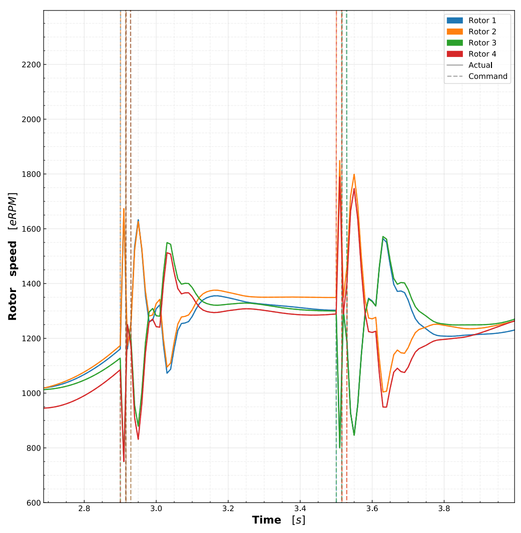


Fig. 126 Rapid oscillations, induced by the PID controller, in the commanded rotor speed of the identified indoor polynomial model of the MetalBeetle which occur during a step input to the attitude angles.

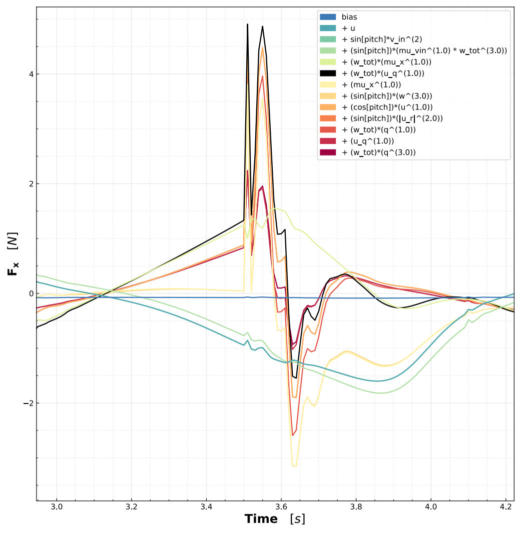


Fig. 127 Cumulative contributions of the constituent regressors of the identified polynomial model of F_x for the indoor MetalBeetle localized around a step input to the attitude angles. Highlighted in black is the regressor which introduces oscillations to the model. Regressor contributions are in descending order in the legend.

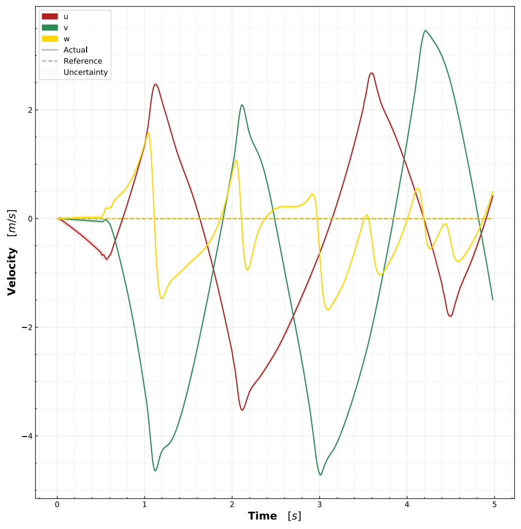


Fig. 128 Velocity response of the identified indoor polynomial model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

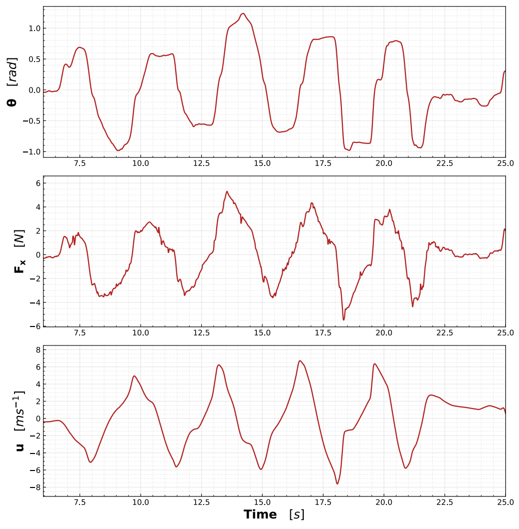


Fig. 129 Indoor flight data of taken from the MetalBeetle depicting the change in force, F_x, and velocity, u, due to oscillating commanded pitch angles, θ.

The associated force and moment responses are depicted in fig. 132 and fig. 133 respectively. From the F_z response, the induced upwards force by the yawing manoeuvre is apparent and aligns well with the controller's tendency to reduce the thrust as discussed for the commanded rotor speeds. These spikes in F_z are short since they correspond to the rapid changes in yaw. Interestingly, the F_y force also exhibits a notable non-zero response. This is likely induced by the coupling of the runaway F_x term and the yaw rotation of the quadrotor. Indeed, as the quadrotor translates with a

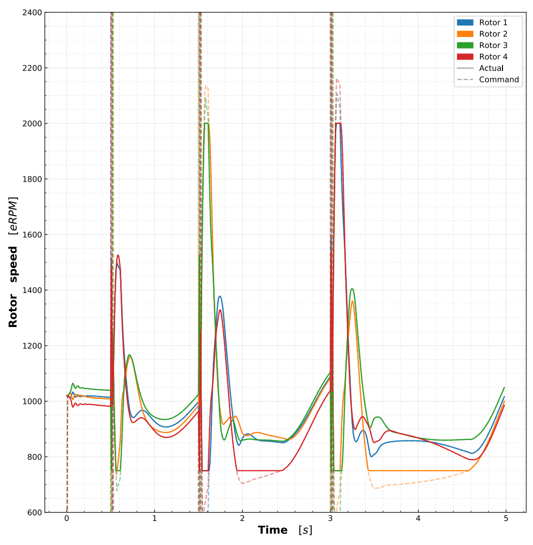


Fig. 130 Commanded rotor speed of the identified indoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

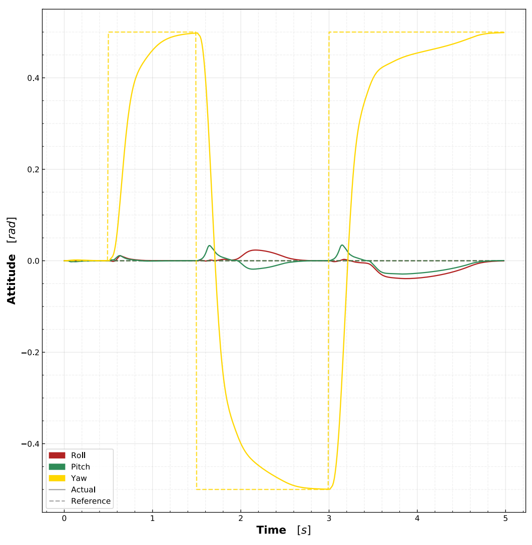


Fig. 131 Attitude response of the identified indoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

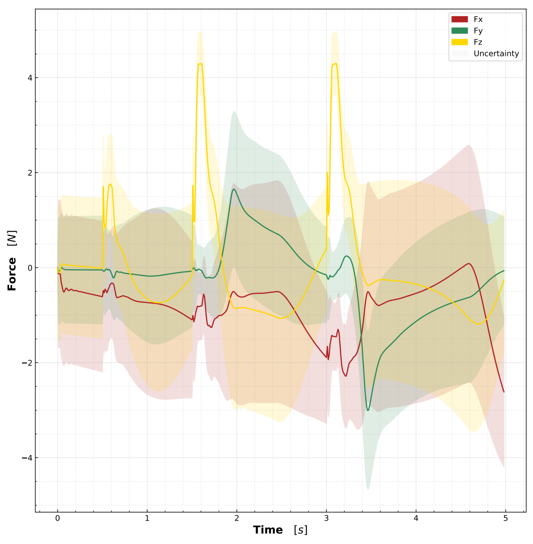


Fig. 132 Force response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

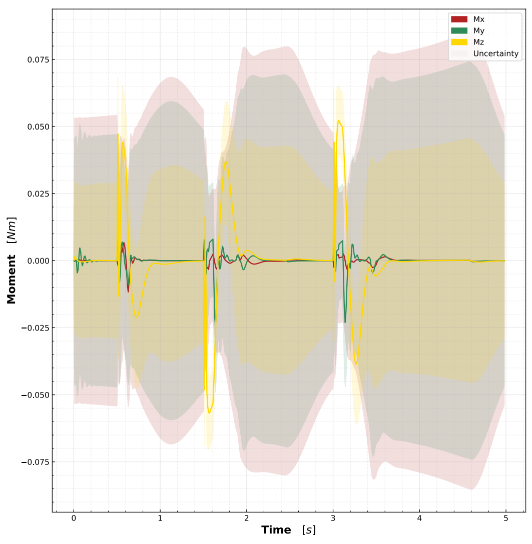


Fig. 133 Moment response, along with associated prediction intervals, of the identified indoor polynomial model of the MetalBeetle when subject to oscillating yaw angle step inputs.

non-zero velocity along x and subsequently rotates along its yaw axis, it will stimulate a velocity along y due to the inertia of the quadrotor. However, due to the runaway velocities, instabilities will occur should they get too large. This is starting to happen in the force response as the magnitudes of F_x and F_y appear to increase in magnitude towards the end of the response. Furthermore, the yaw moment spikes align with the yaw step transitions and consist of an initial peak followed by an opposite peak to stabilize the angle. Such a response is reflective of physical principles and the response of the true system as shown in fig. 134. Again, the moment response PIs display a caterpillar-like shape whereby uncertainties are high for regions of no excitation and low for those where non-zero moments occur. Importantly, the

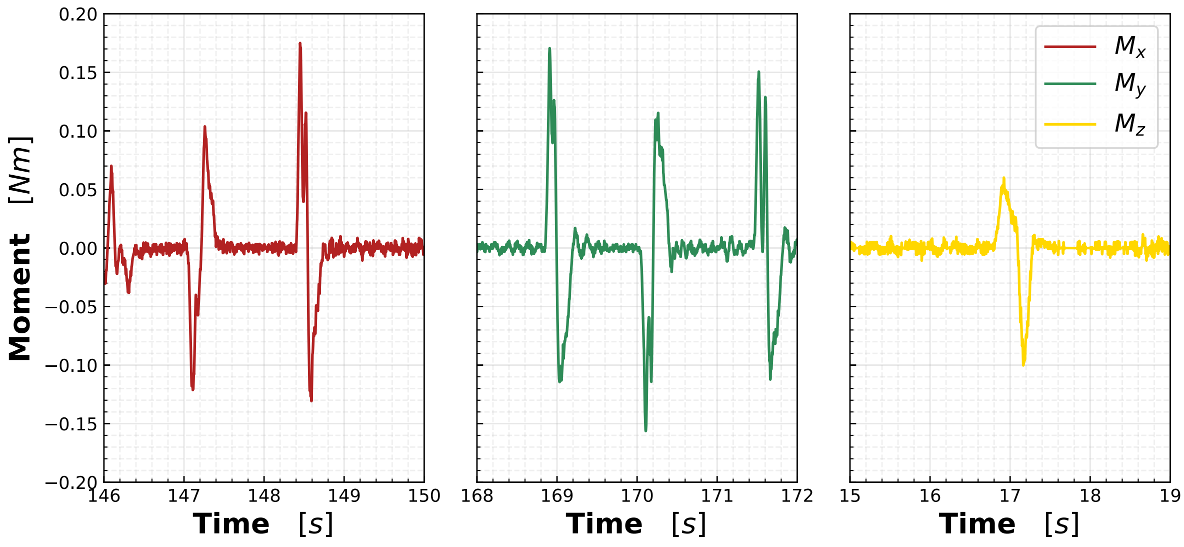


Fig. 134 Example of the measured moments during attitude step inputs in the identification data of the indoor MetalBeetle.

PIs increase over time and perhaps forecast some instability in the model due to the growing runaway forces. Hence, aside from the runaway F_x term, the force and moment responses to the yaw step responses appear reasonable.

Hybrid model

As the hybrid indoor MetalBeetle models build upon the corresponding polynomial models, and are not seen to significantly improve model performance, they are expected to demonstrate similar simulation responses. However, this is not the case for hovering flight where even more instabilities can be seen as the hybrid response diverges considerably from that of the polynomial model.

The corresponding commanded rotor speeds and resultant attitude are shown fig. 135 and fig. 136 respectively. Unlike the polynomial models' response where the rotor speeds are relatively constant, the hybrid rotor speeds slowly, and unrealistically, fan-out. Such behaviour is caused by the ANN compensator since the underlying polynomial produces the correct hovering response. Although the exact reasons for the divergence of rotor speeds are unclear, it is suspected that the hovering states, and runaway velocity, cause the ANN compensator to produce erroneous predictions. Consequently, the expansion of the rotor speeds are, in effect, due to the controller responding to misbehaving moment models. This perceived input mismatch only pushes the ANN component to make further mistaken predictions which cause the rotors to diverge more. Indeed, the controller is able to maintain hover until the lower rotor saturation limit is reached by rotor 1, after which instabilities occur. It is at this point (around 1.4 seconds) that the roll and pitch attitude angles begin to diverge from zero as seen in fig. 136. Ironically, the ANN compensator designed to handle such unknown situations deteriorates performance in one such scenario. As the attitude diverge, and rotor 1 remains saturated, the quadrotor loses control and tends towards instability. This is seen by the rapid oscillations of the commanded rotor speeds after 2.5 seconds wherein the controller frantically attempts to maintain control. Since the hovering states are apparently the issue, a trimmed condition based on the average values of the identification data was used in place of the hovering condition. However, this also experienced similar instabilities due to persistent runaway velocities. Interestingly, the input weight magnitudes associated with the hybrid models forecast this issue as they are found to be mostly reliant on the velocities and attitude angles.

The model instabilities following the saturation of rotor 1 are also apparent in both the force and moment responses of the indoor MetalBeetle hybrid model. These responses are illustrated in fig. 137 and fig. 138 respectively. Again, the runaway F_x term is present although the ANN compensator appears to heavily dampen the associated instability. Even though the hybrid model is objectively worse, this restriction of the runaway F_x is nonetheless desirable and shows promise for the method. Interestingly, the runaway F_x is now positive instead of negative, highlighting the sensitivity of this instability. Up until the saturation of rotor 1, the remaining force and moment models tend to zero after the initial

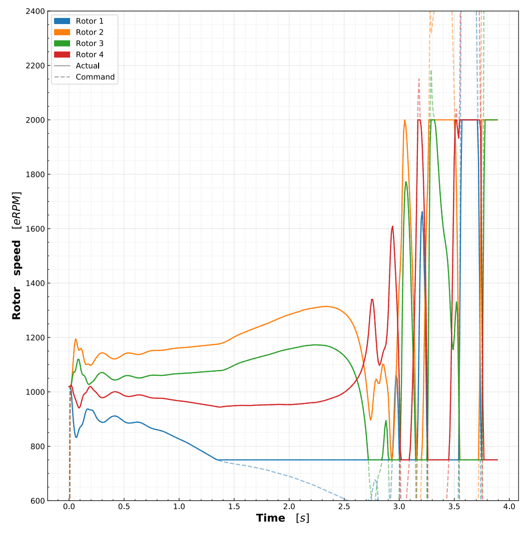


Fig. 135 Commanded rotor speed of the identified indoor hybrid model of the MetalBeetle during hovering flight.

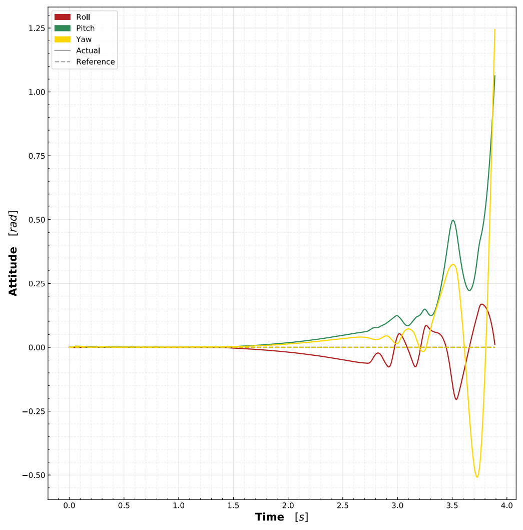


Fig. 136 Attitude response of the identified indoor hybrid model of the MetalBeetle during hovering flight.

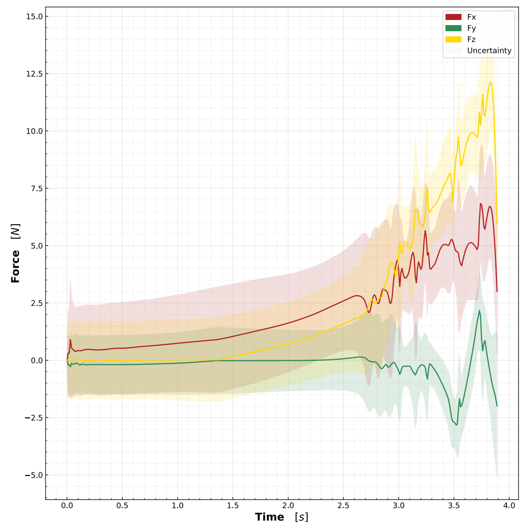


Fig. 137 Force response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle during hovering flight.

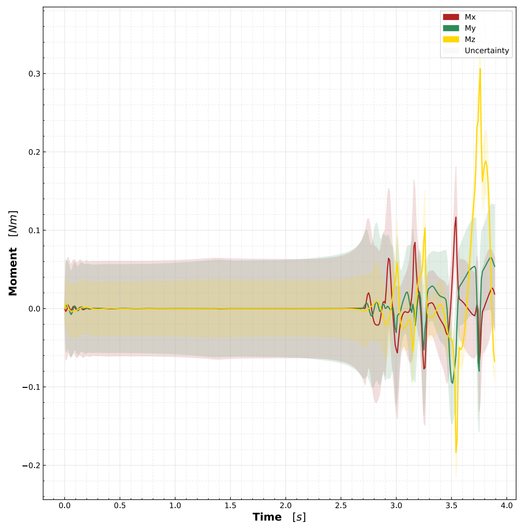


Fig. 138 Moment response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle during hovering flight.

stabilization of the quadrotor. This is in-line with the underlying polynomial model response and is reflective of the expected dynamics during hovering flight. After the rotor saturation, the F_z force begins to destabilize and contributes to the loss of control of the MetalBeetle. Although slower to react, the moment models also tend towards instability around a second after rotor saturation. Again, the associated PIs demonstrate their utility - particularly for the moment models - as they increase during the initial rotor saturation and continue to do so until the instabilities are prominent. As such, the PIs may be used to forecast such stability issues.

The hybrid models response is simulated for combined roll & pitch oscillating step inputs and step inputs into the yaw for a more representative evaluation of the model's capacity around regions of the flight envelope for which it is valid. These inputs are similar to those used in the corresponding polynomial model simulation, but differ slightly in order to confine the states to the domain of validity.

The commanded rotor speeds and resultant attitude responses of the indoor MetalBeetle hybrid model with respect to oscillating roll and pitch inputs are illustrated by fig. 139 and fig. 140 respectively. Once again, the commanded rotor speeds commence their divergence prior to any of the manoeuvres. However, during the step inputs, these rotor speeds are more realistic and compressed. This supports the hypothesis that the expansion of rotor speeds during hover or trimmed flight is due to the unstable ANN predictions on unknown regions of the flight envelope. Moreover, the average rotor speed also increases for non-zero attitudes and reflects the additional thrust necessary to maintain altitude. Indeed, the quadrotor is able to track the roll and pitch step inputs effectively and realistically. Furthermore, the commanded rotor speeds are reminiscent of the underlying polynomial model.

The subsequent force response of indoor MetalBeetle hybrid model is depicted in fig. 141. Overall, the force responses reflect expectations, mimicking the underlying polynomial responses, but exhibit some curious and unrealistic force instabilities. The triangular nature of the F_x and F_y force responses parallels the measurement data of a similar manoeuvre shown in fig. 129, albeit with more oscillations. Indeed, slight differences in the measurement data are expected since the throttle was increased to expedite direction changes and acceleration during the flight tests, but is not conducted in the simulation for simplicity. The ANN compensator of hybrid model appears to dampen out much of the sharp oscillations seen in the corresponding polynomial response and subsequently gives a more realistic representation of the force response. This is evident through fig. 144, which illustrates the cumulative contributions - in descending order of the legend - of the underlying polynomial regressors of F_y. Most of the oscillations appear to be introduced by the total rotor speed, $\bar{\omega}_{tot}$ in the $\bar{\omega}_{tot}\mu_y$ term instead of the control moments. Again, these effects are a byproduct of the rapid rotor oscillations in the sub-optimal PID controller. Interestingly, the contribution of $\bar{\omega}_{tot}\mu_y$ is remarkably similar to the the measured forces, and highlights the necessity of this term and velocity in general. Peculiarly, the F_y force exhibits a growing non-zero force following the pitch step input, which is unrealistic. Recall that the ANN compensator is found to rely most significantly on $\cos(\theta)$ (refer to fig. 108). As such, the ANN compensator likely initiates an F_y response towards a pitch change, after which the polynomial model amplifies the erroneous force. While the F_x response is mostly reasonable, the effects of the runaway velocity are still apparent in the reluctance of the F_x to increase its force towards the positive but apparent enthusiasm for accelerating towards the negative. These F_x and F_y issues are obvious in the velocity response of the hybrid model fig. 143. Again, the shapes of these responses reflect the measurement data well. However, overall, the hybrid force models are considered invalid as they produce unreliable force changes enticed by the ANN compensator.

In contrast, the associated moment responses presented in fig. 142 appear to be sensible. Sharp peaks in M_x and M_y occur and correspond to the attitude changes while the M_z response is largely zero. As with the measurement data

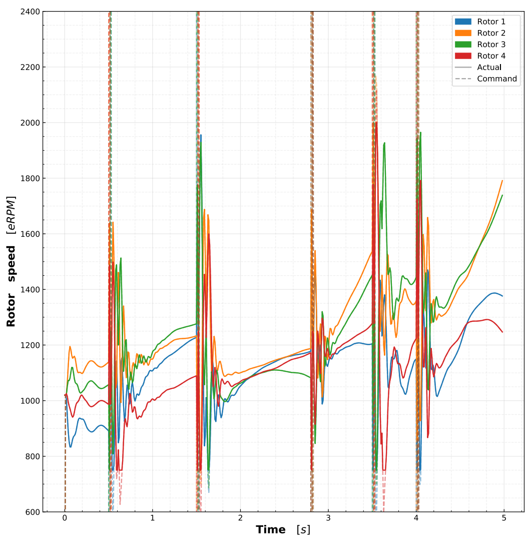


Fig. 139 Commanded rotor speed of the identified indoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

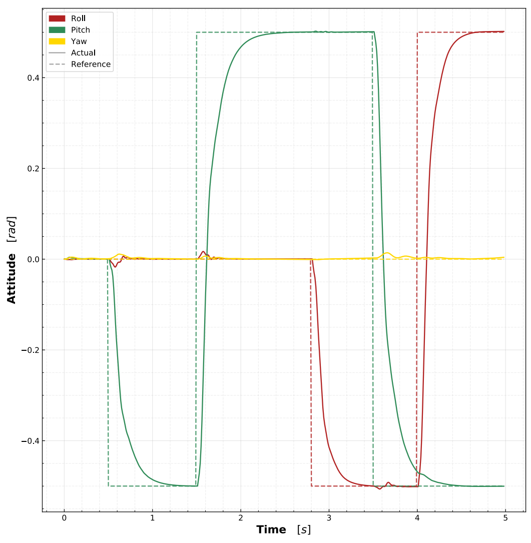


Fig. 140 Attitude response of the identified indoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

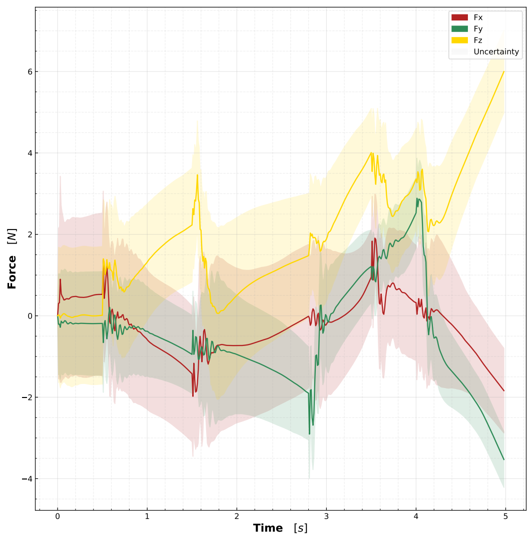


Fig. 141 Force response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

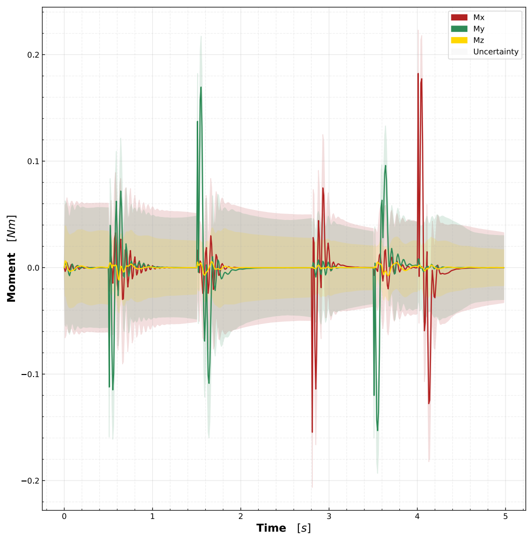


Fig. 142 Moment response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle subject to oscillating roll and pitch step inputs.

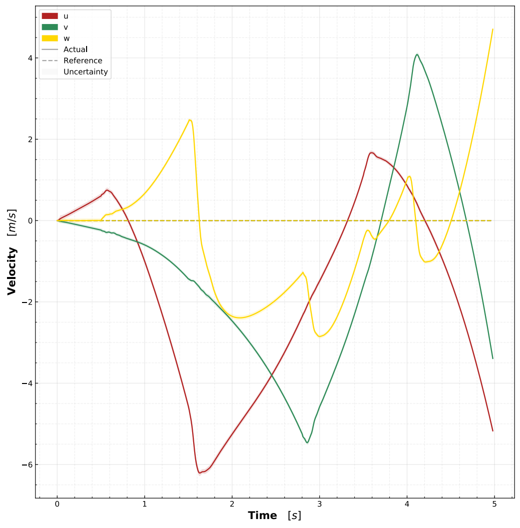


Fig. 143 Velocity response of the identified indoor hybrid model of the MetalBeetle when subject to oscillating roll and pitch angle step inputs.

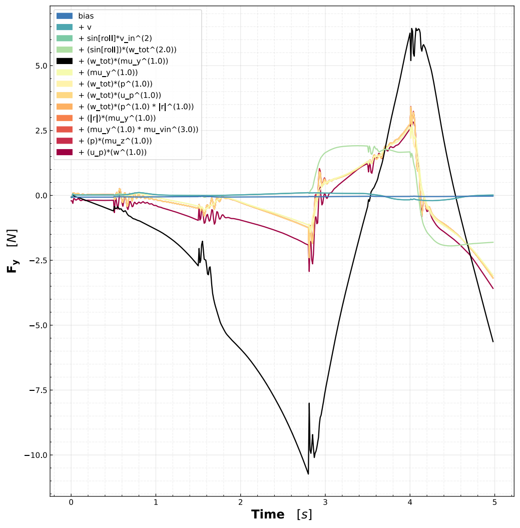


Fig. 144 Cumulative contributions of the constituent regressors of the polynomial componentn of the identified hybrid model of F_y for the indoor MetalBeetle subject to oscillating roll and pitch angle step inputs. Highlighted in black is the regressor that introduces oscillations. Regressor contributions are in descending order in the legend.

fig. 134, the moment responses show an initial peak and subsequent correction (i.e. breaking) in the opposing direction to stop the rotation change. The high-frequency oscillations of the M_x or M_y response during and surrounding the peak are likely due to the sub-optimal PID controller in tandem with saturated rotors. The associated PIs are narrowest during a response, implying model confidence for such manoeuvres.

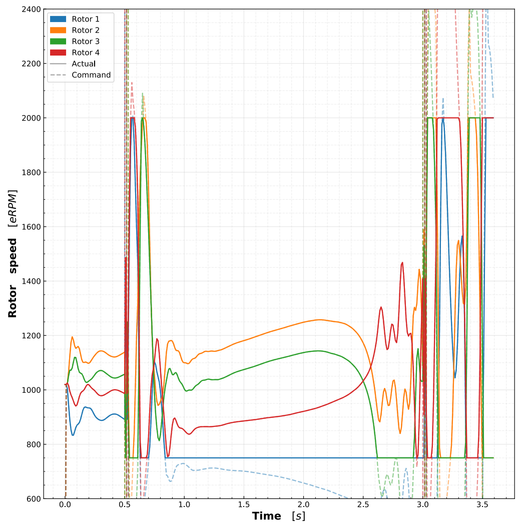


Fig. 145 Commanded rotor speed of the identified indoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

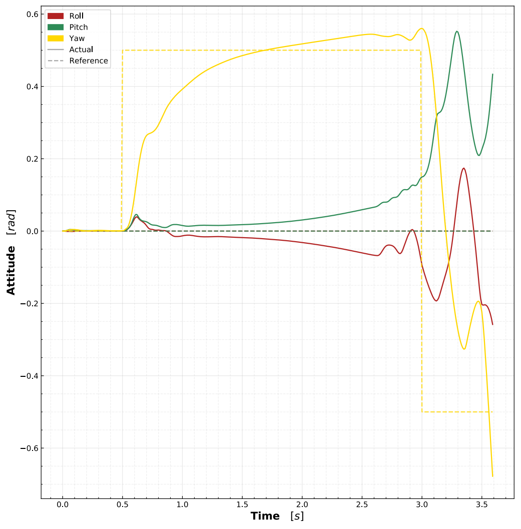


Fig. 146 Attitude response of the identified indoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

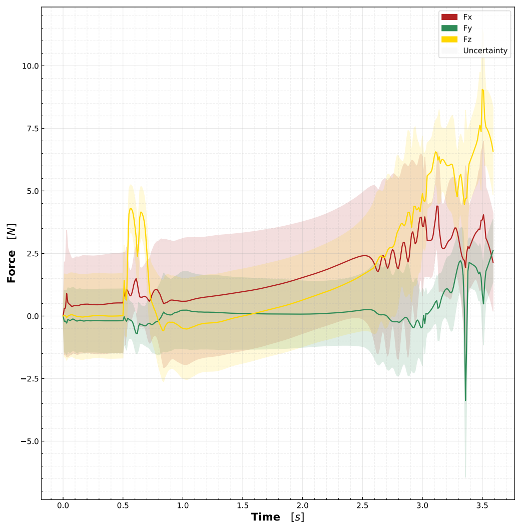


Fig. 147 Force response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

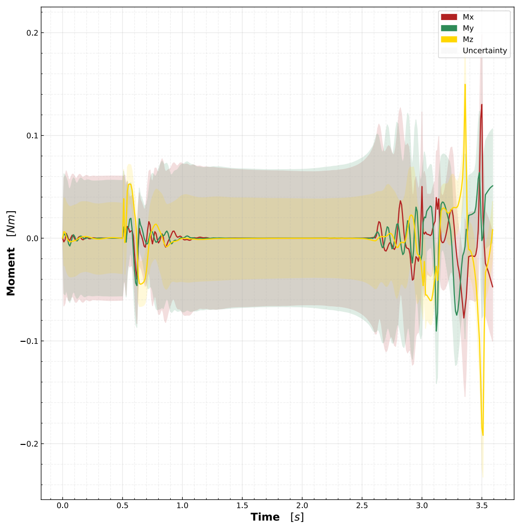


Fig. 148 Moment response, along with associated prediction intervals, of the identified indoor hybrid model of the MetalBeetle subject to oscillating yaw step inputs.

The corresponding rotor commands and attitude for the step yaw simulations are illustrated in fig. 145 and fig. 146 respectively. It is immediately clear from the commanded rotor speeds that the quadrotor has lost control during the yaw step inputs. The attitude responses reveal that the hybrid model struggles to track the initial yaw step input through a slow response and overshoot. Subsequently, it fails entirely to follow the second step input. This is likely due to the increase pitch and roll angles throughout the manoeuvre. Such increases are unrealistic and stem from the ANN components of the hybrid models. As with the hover case, one of the rotors becomes saturated after the first yaw manoeuvre, leading to an uncontrollable attitude. Thus similar instabilities, which propagate from the diverging attitudes, are observed in the step yaw response. Note that, instead of an oscillating yaw input, a single step input to the yaw was also simulated.

Nevertheless, the instability issues persisted as they originate from the initial step response.

The corresponding force and moment responses for the oscillating yaw step inputs are visualized in fig. 147 and fig. 148 respectively. As instabilities are incited in a similar manner to the hovering case (i.e. through diverging attitudes), the subsequent force and moment responses share many of the same characteristics. Indeed, both F_x and F_z begin their runaway upon the saturation of rotor 1. The moments then show delayed oscillations, in accordance with the oscillating rotor speeds, which increase over time. As with the underlying polynomial model, the F_z increases in tandem with the step yaw input due to a net increase in average rotor speeds in comparison to the speeds needed for hovering flight. After this spike in F_z, the rotor speeds return to their idle states and thus F_z reduces. Likewise, the sine-wave like shape of the M_z response during this manoeuvre is sensible and reflective of the measurement data for similar inputs as shown in fig. 134.

However, due to the obvious instabilities of the hybrid model, it can not be considered valid or useful for simulation. Nevertheless, there are some merits to using the hybrid approach as seen in the damping of the polynomial runaway F_x. Consequently, future research should investigate the causes of the ANN instabilities and explore methods or alternatives for reducing the erroneous predictions.

Artificial Neural Network model

The simulation results of the hybrid indoor MetalBeetle model anticipate poor performance by the ANN-only models, given that many of the observed instabilities are suspected to originate from the ANN components. Unfortunately, this is indeed the case for the simulations of the indoor MetalBeetle ANN model, even with the unconverged F_z ANNs removed from the ensemble. Nonetheless, they are still discussed here with potential reasons for the unrealistic responses explained.

First, the indoor MetalBeetle ANN model is subject to a hovering simulation to modify the PID controller and investigate any inherent instabilities. The associated commanded rotor speeds are depicted in fig. 149. From the conspicuous saturation of rotor speeds, it is obvious that the ANN model has quickly lost control. The high-frequency initial oscillations in the response result from too high derivative (D) gains in the PID controller which, when decreased, do reduce the magnitude of these oscillations. However, this reduction leads to low frequency oscillations in the attitude response. It is therefore a trade-off between high-frequency rotor oscillations and low-frequency response oscillations. Neither are desirable and both are indicative of an overly sensitive model. In an effort to improve the response, the simulation was also run at 500 Hz[20], but the instability issues persisted. Similar to the hybrid models, a trimmed flight condition based on the identification data was also used to verify if stability could be maintained, but the ANN

[20] 500 Hz is the sampling frequency of the data used for model identification.

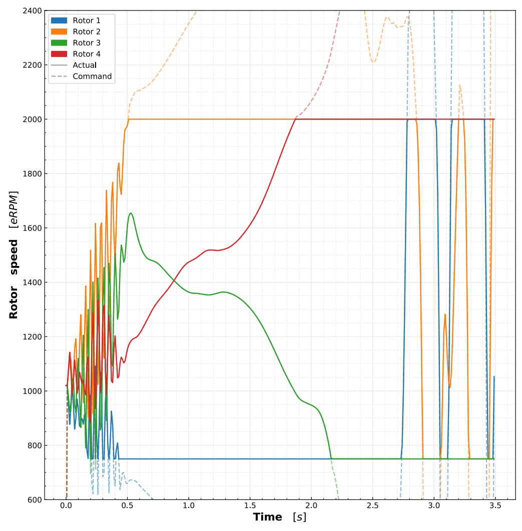


Fig. 149 Commanded rotor speed of the identified indoor ANN model of the MetalBeetle during hovering flight.

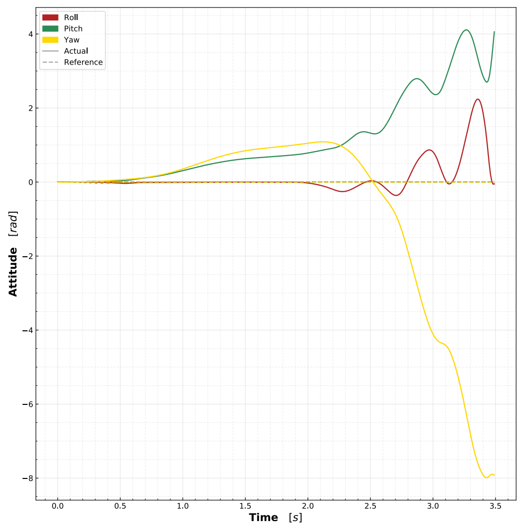


Fig. 150 Attitude response of the identified indoor ANN model of the MetalBeetle during hovering flight.

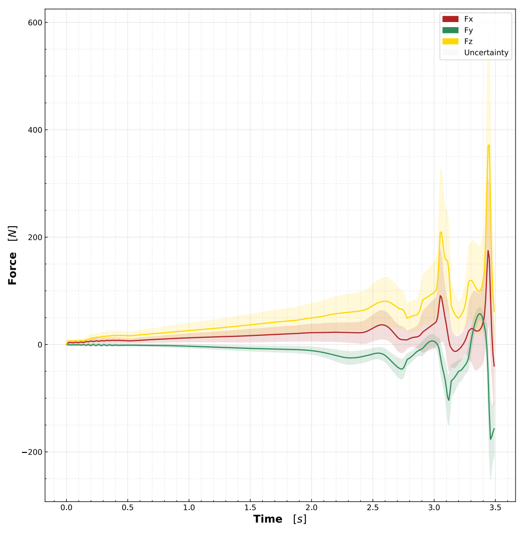


Fig. 151 Force response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle during hovering flight.

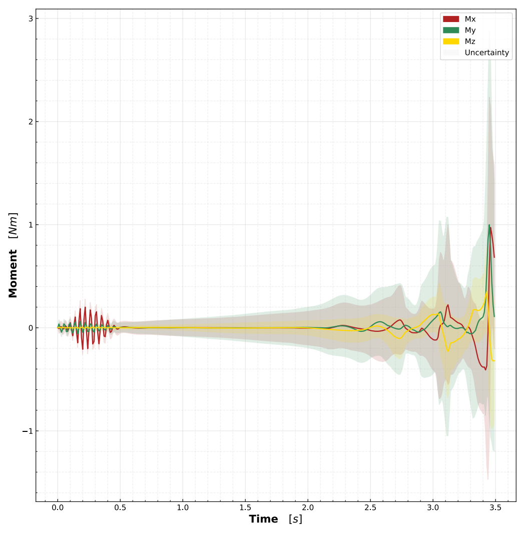


Fig. 152 Moment response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle during hovering flight.

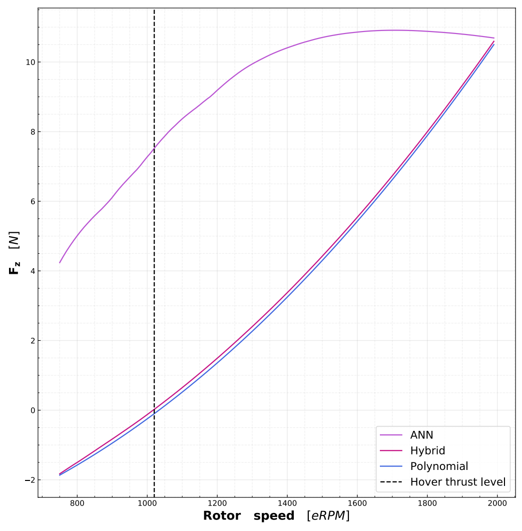


Fig. 153 Thrust profile, as a function of the average rotor speed, of the identified polynomial-only, ANN-only, and Hybrid models of the indoor MetalBeetle. Here, zero corresponds to the thrust necessary to maintain hover. As such, also shown is the true hovering rotor speed of the MetalBeetle. Note that this profile is generated with all other states set to zero.

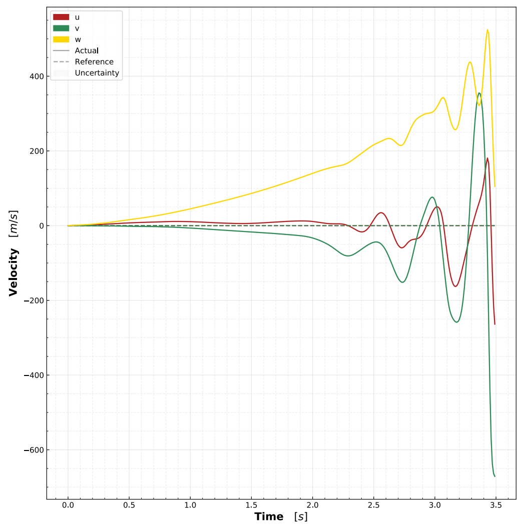


Fig. 154 Velocity response of the identified indoor ANN model of the MetalBeetle during hovering flight.

models nonetheless lost control. The reasons for this are related to the ANN F_z and are discussed in more depth when addressing the associated force response. The resultant attitude response is shown in fig. 150 for which the loss of control is apparent through the growing and oscillating attitude angles. These instabilities appear to begin as soon as rotor saturation is reached and is therefore reminiscent of the hybrid model instabilities.

The ANN model instability is also apparent in the corresponding force and moment responses which are illustrated

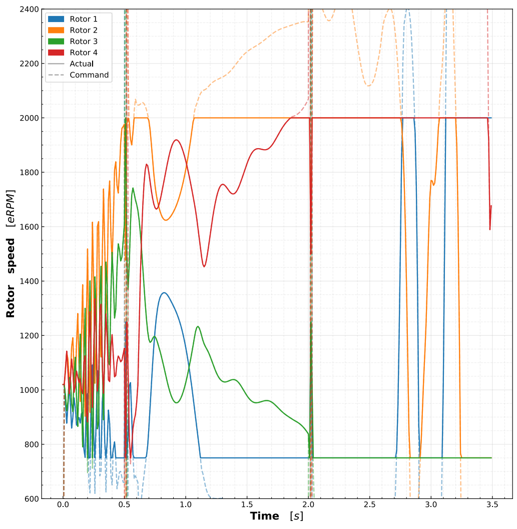


Fig. 155 Commanded rotor speed of the identified indoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

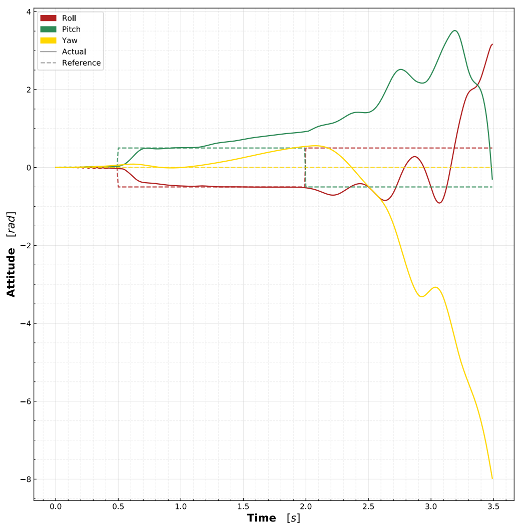


Fig. 156 Attitude response of the identified indoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

in fig. 151 and fig. 152 respectively. The force response especially blows up to such a degree that it is already invalid before rotor saturation occurs. Consequently, some of the indoor MetalBeetle ANN model's issues may propagate force models. The primary offender is the F_z model, which is found to be unrealistic even with the unconverged ANNs removed. Figure 153 depicts the thrust profiles of the identified ANN-only, polynomial-only, and hybrid models of the indoor MetalBeetle as a function of the rotor speed with all other states set to zero. Also shown is the true rotor speeds corresponding to the hovering flight of the MetalBeetle observed during data acquisition. While the polynomial-based thrust profiles are somewhat linear[21] and match the hovering thrust rotor speed well, the ANN-only thrust profile is always significantly above zero regardless of the rotor speed. As such it is always accelerating upwards with over $1g$ of thrust (recall the mass of the MetalBeetle is $0.396\ kg \approx 4\ N$) and thus quickly leads to instabilities. In fact, due to this, the altitude control is turned off during the ANN simulations to promote a longer period of stability. However, as the velocity along z, w, quickly rises, it subsequently propagates the instabilities to the remaining models and states. Figure 154 illustrates the associated velocity response of the quadrotor during hover wherein the runaway w can be seen. Recall that due to the dense nature of these ANNs, runaway inputs will dominate the models' predictions at some point, no matter how low the significance of the input is[22]. Consequently, the other velocities also begin to diverge from zero. Therefore, this poor F_z model is a contributing factor, if not the root, of the problems which plague the identified indoor MetalBeetle ANN models. As such, the identified ANN model is invalid in the hovering condition.

Indeed, the hover condition is outside the identification envelope of the ANN models, and may therefore explain its poor stability. Consequently, it is subjected to oscillating roll and pitch step inputs similar to what is seen in the training data. Despite this, due to the unrealistic F_z ANN model, the resultant response tends towards instability.

The associated commanded rotor speeds are illustrated in fig. 155 and the subsequent attitude response is presented in fig. 156. Similar to the hover case, the commanded rotor speeds exhibit rapid oscillations during the initial stabilization phase of the quadrotor. These quickly saturate but control is somewhat maintained through the commanded step inputs which cause the rotor speeds to return from their saturation limits. The attitude response reveals that the ANN model is initially able to track the commanded step inputs to pitch and roll. However, this does not last as the rotor speeds soon saturate again and the attitudes begin diverging from their references. Subsequently, the ANN model is unable to recover tracking for the next step inputs and tends towards instability. Given that control was initially maintained through the onset of the step inputs, it may be argued that perhaps with rapid enough step inputs, control of the ANN model may be maintained before any of the forces runaway. Even if this may be the case, the resultant model is still unrealistic and invalid. Furthermore, there is no check on the increasing F_z which, due to the dense nature of the ANNs,

[21]This is useful for control since the PID controller is a linear controller, and thus works best with such linear responses.
[22]Unless, of course, all the associated weights are at zero. However, this does not happen in the identified ANN models.

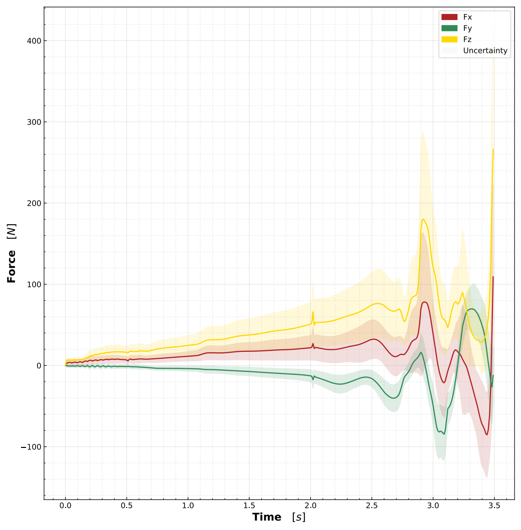


Fig. 157 Force response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

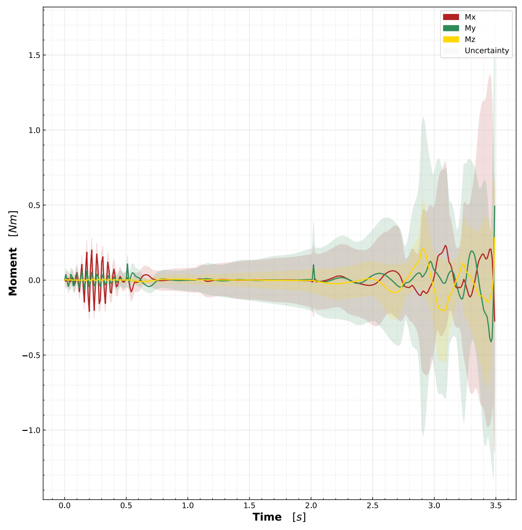


Fig. 158 Moment response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle subject to oscillating roll and pitch step inputs.

will ultimately cause issues for the remaining models.

Indeed, the force response depicted in fig. 157 show that these step inputs do not recover the runaway forces. In fact, during the step responses, all forces only increase in magnitude beyond the domain of validity of the model. As anticipated, the most egregious offender is F_z, which holds the highest magnitudes and is likely the root of the model instabilities. Likewise, the moment responses tend towards instability as shown in fig. 158. Note that, during the step response, the identified moments do indeed show the expected response. Hence, there is some validity to the identified moment models. However, this is overshadowed by the subsequent instability. What is perhaps conducted well in the ANN models is the evident increase of the PI widths as the model veers towards instability. Such an increase is especially pronounced for the moment models. This is reflective of the models', undoubtedly well-placed, uncertainty in its predictions and once again highlights the utility of these intervals.

To complete the attitude response, the indoor MetalBeetle ANN models may be subject to step inputs into the yaw angle. Unsurprisingly, the ANN model is unable to maintain stability in the simulation of such inputs.

The associated commanded rotor speeds are presented in fig. 159 with the corresponding attitude response given by fig. 160. Consistent with prior simulations of the ANN indoor MetalBeetle, the commanded speeds are largely saturated and fail to control the quadrotor. From the attitude response, it is clear that the quadrotor fails to adequately follow even the first yaw step input by overshooting significantly. Before the second yaw step input, the model has already reached an unstable state and is therefore unable to track the reference yaw angle.

The corresponding force and moment responses also closely resemble their counterparts in previous simulations of the ANN indoor MetalBeetle. These responses are depicted in fig. 161 and fig. 162 respectively. This inter-simulation similarity highlights the inherent instability of the ANN model and difficulties faced in trying to maintain control. Briefly, the force responses again increase in magnitude rapidly, rallied on by the unfeasible F_z model. The moment responses show initial oscillations associated with the stabilization of the quadrotor (i.e. before rotor saturation) and a subsequent tranquil period before oscillating towards instability themselves. The associated PIs increase as the instabilities grow, accentuating the models' poor confidence with the predictions during this phase.

Overall, it is clear that, despite the apparently good performance metrics, the identified ANN model of the indoor MetalBeetle is invalid. In particular, due to the unrealistic F_z model. While there are some traces of validity in the remaining constituent ANN models, such as the moment responses, the aggregate model does not provide much utility. What may perhaps be an interesting avenue of further research is to investigate the effects of a hybrid approach which employs different techniques for the model constituents. For example, by replacing the ANN F_z model here with its polynomial counterpart, perhaps performance may be improved.

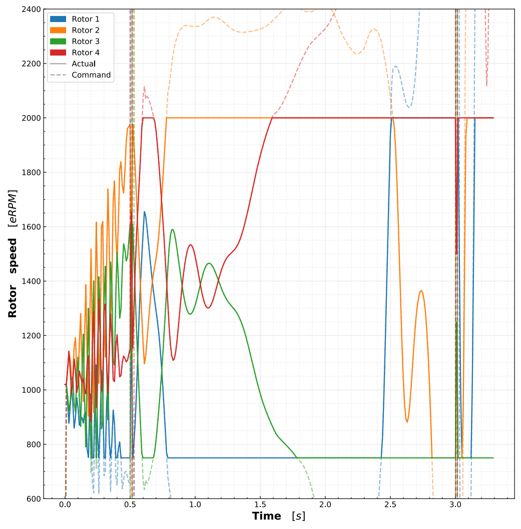


Fig. 159 Commanded rotor speed of the identified indoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

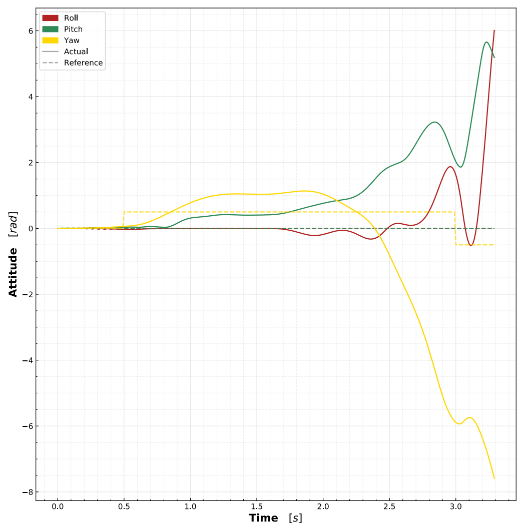


Fig. 160 Attitude response of the identified indoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

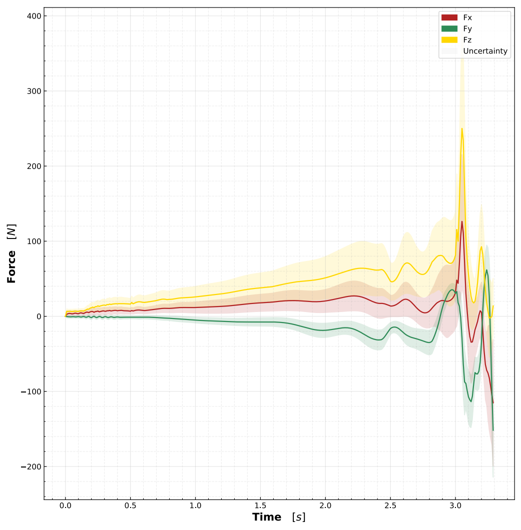


Fig. 161 Force response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

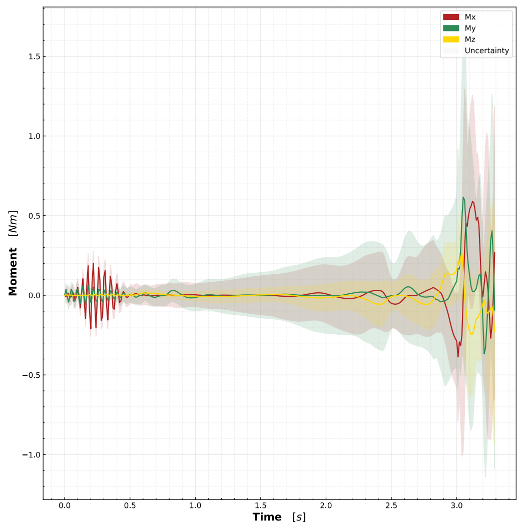


Fig. 162 Moment response, along with associated prediction intervals, of the identified indoor ANN model of the MetalBeetle subject to oscillating yaw step inputs.

Summary of indoor MetalBeetle simulations

In summary, the simulations of the identified models of the indoor MetalBeetle show that the polynomial models consistently produces the most feasible and useful models. The use of ANNs appears to introduce instabilities into the model primarily due to high magnitude inputs which propagate throughout the remaining, perhaps even valid, models. In the hybrid case, the runaway F_x of the polynomial model misleads the ANN compensator predictions. For the ANN-only models, the poor F_z constituent models produces too much thrust which incites instabilities. Nonetheless, the runaway F_x is a cause for concern and ultimately invalidates the polynomial models as well.

5. Indoor MetalBeetle Results: Extra

In this subsection, additional tables and plots associated with the identification results of the indoor MetalBeetle are shown.

Plots of model predictions

In this subsection, complementary plots of the force and moment predictions of unseen (i.e. validation) flights for the indoor MetalBeetle are shown. In particular, the remaining force and moment plots of the validation flight which are not illustrated in section VIII.D.2.

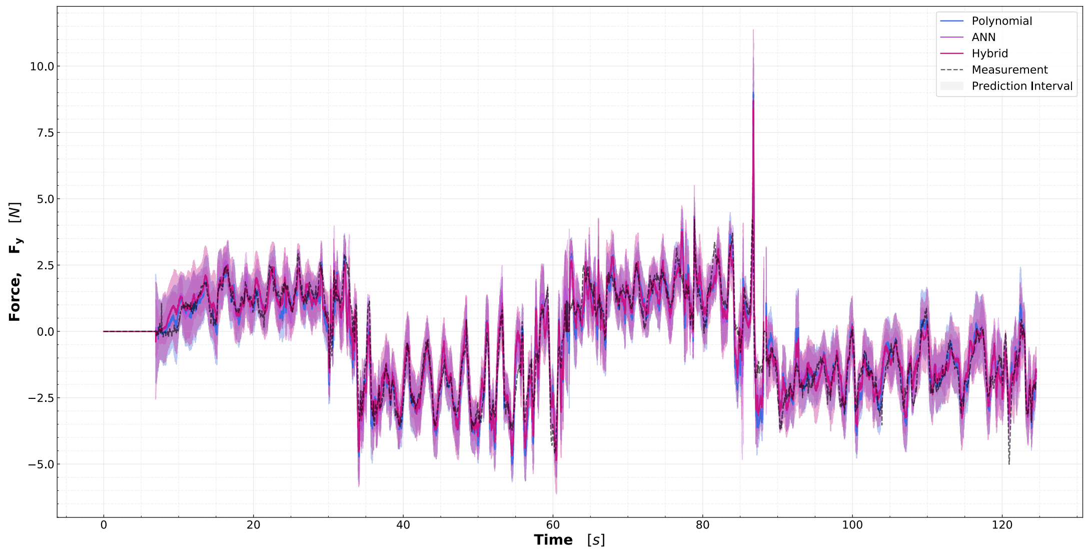


Fig. 163 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_y for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured F_y response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

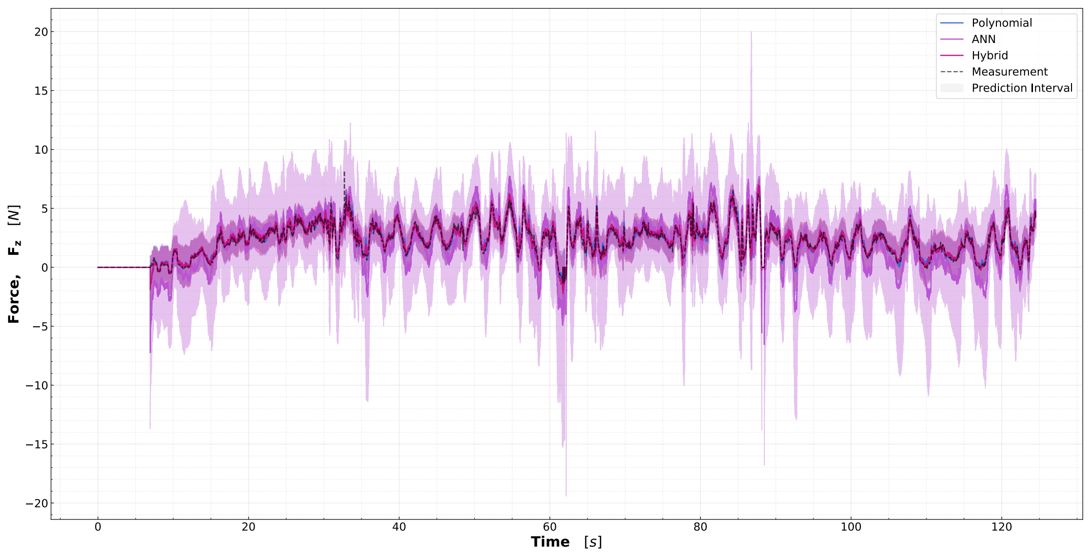


Fig. 164 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of F_z for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured F_z response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

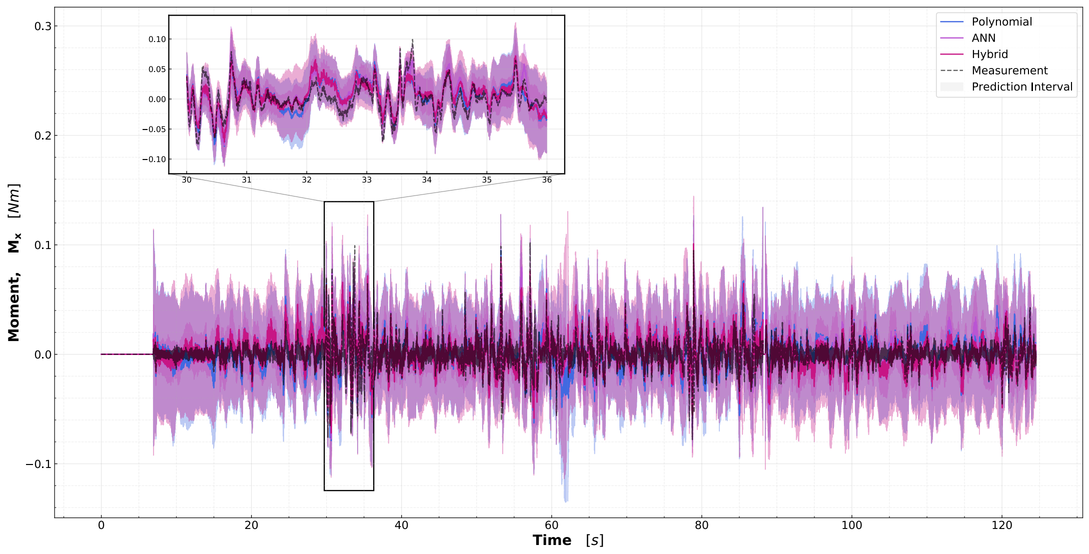


Fig. 165 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_x for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured M_x response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

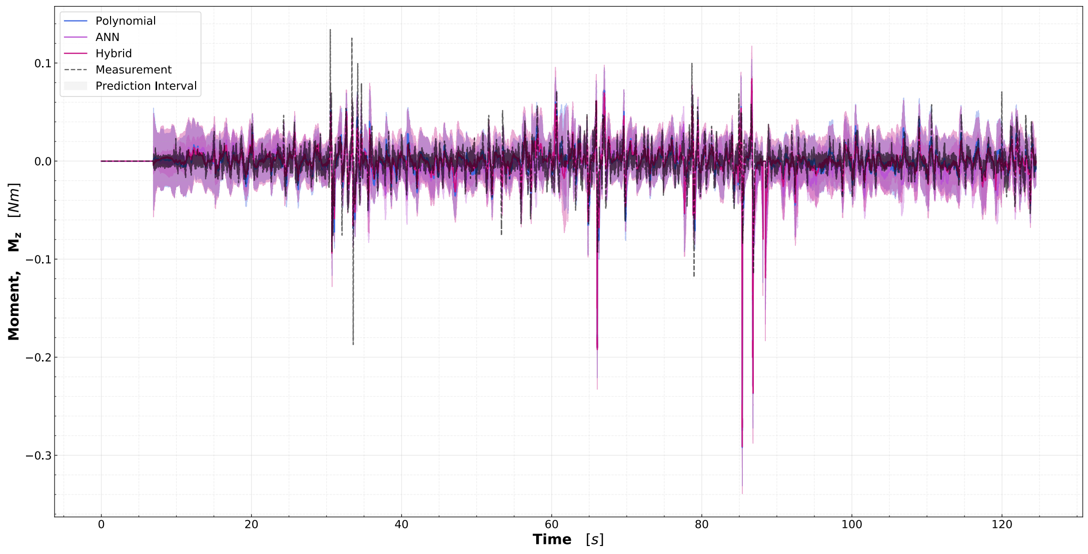


Fig. 166 Comparison of the identified ANN-only (in purple), Polynomial-only (in blue), and Hybrid (in magenta) models predictions of M_z for flight 24 (indoor untrained flight) of the MetalBeetle. Also shown is the measured M_z response (black dotted line). Flight 24 is an first-person-view flight and involves flying around a flag pole at the center of the CyberZoo.

Identified polynomial model structures

In this subsection, the regressors of the identified polynomial models of the indoor MetalBeetle are summarized. Descending down the tables gives the order of selection. Also shown are the associated covariances and cumulative accuracy contribution of each regressor.

Table 15 Identified polynomial model of C_x for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	-2.337E-02	0.0009	0.0000
$\bar{u}$	-4.356E-01	0.0036	0.0044
$\sin(\theta)\bar{v}_{in}^2$	1.008E-06	0.0004	0.0099
$\sin(\theta)\mu_{v_{in}}\bar{\omega}_{tot}^3$	3.018E+01	0.0028	0.8168
$\bar{\omega}_{tot}\mu_x$	5.748E+00	0.0105	0.8605
$\bar{\omega}_{tot}\bar{u}_q$	3.453E+00	0.0258	0.8855
μ_x	-1.010E+00	0.0032	0.9063
$\sin(\theta)\bar{w}^3$	2.848E-01	0.0014	0.9138
$\cos(\theta)\bar{u}$	4.795E-01	0.0036	0.9237
$\sin(\theta)\lvert\bar{u}_r\rvert^2$	1.084E-01	0.0008	0.9288
$\bar{\omega}_{tot}\bar{q}$	1.043E+01	0.0996	0.9330
$\bar{u}_q$	-5.269E-01	0.0084	0.9359
$\bar{\omega}_{tot}\bar{q}$	-2.207E+02	8.3258	0.9372

Table 16 Identified polynomial model of C_y for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	-1.228E-02	0.0006	0.0000
$\bar{v}$	-4.201E-02	0.0017	0.1002
$\sin(\phi)\bar{v}_{in}^2$	-1.029E-05	0.0001	0.1128
$\sin(\phi)\bar{\omega}_{tot}^2$	-1.569E+01	0.0007	0.8282
$\bar{\omega}_{tot}\mu_y$	9.358E+00	0.0071	0.9077
μ_y	-2.004E+00	0.0030	0.9334
$\bar{\omega}_{tot}\bar{p}$	-4.942E+00	0.0531	0.9435
$\bar{\omega}_{tot}\bar{u}_p$	1.075E+00	0.0037	0.9500
$\bar{\omega}_{tot}\bar{p}\lvert\bar{r}\rvert$	-1.925E+02	0.9175	0.9532
$\lvert\bar{r}\rvert\mu_y$	-2.537E+00	0.0417	0.9574
$\mu_y\mu_{v_{in}}^3$	1.617E-02	0.0001	0.9594
$\bar{p}\mu_z$	1.489E+00	0.0213	0.9611
$\bar{u}_p\bar{w}$	-1.595E-01	0.0032	0.9623

Table 17 Identified polynomial model of C_z for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	R^2
$bias$	-7.557E-03	0.0114	0.0000
$\bar{\omega}_{tot}^2$	1.488E+01	0.0225	0.7039
$(\bar{u}^2 + \bar{v}^2)$	2.895E-02	0.0035	0.7068
$(\bar{v}_{in} - \bar{w})^2$	-2.588E-08	0.0000	0.7181
$\bar{w}$	-1.725E-02	0.0015	0.7190
$\cos(\theta)\bar{\omega}_{tot}^2$	-1.568E+01	0.0079	0.7288
$\cos(\phi)\bar{\omega}_{tot}^2$	-1.471E+01	0.0115	0.7458
$\bar{\omega}_{tot}\lvert\bar{u}_q\rvert\lvert\bar{u}_r\rvert$	2.408E-01	0.1554	0.7495
$\bar{\omega}_{tot}^4$	3.347E+02	0.0173	0.9738
$\bar{\omega}_{tot}\lvert\bar{u}_r\rvert^2$	2.074E+00	0.0168	0.9753
$\bar{\omega}_{tot}\lvert\mu_x\rvert\mu_z$	7.475E-01	0.0091	0.9766
$\lvert\bar{u}_r\rvert^2$	-3.408E-01	0.0052	0.9777
$\bar{\omega}_{tot}\lvert\bar{u}_q\rvert\lvert\bar{u}_p\rvert$	8.767E+00	0.1634	0.9781
$\lvert\bar{u}_q\rvert\lvert\bar{u}_p\rvert$	-1.388E+00	0.0382	0.9785

Table 18 Identified polynomial model of C_l for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	$\mathbf{R^2}$
$bias$	-2.055E-06	0.0001	0.0000
$\bar{p}$	-1.053E-02	0.0010	0.0002
$\bar{u}_p$	-4.165E-03	0.0001	0.4125
$\bar{\omega}_{tot}\bar{u}_p$	2.725E-02	0.0001	0.5645
$\bar{\omega}_{tot}\mu_y$	-1.348E-03	0.0000	0.5993
$\bar{\omega}_{tot}\bar{u}_p\lvert\bar{u}_r\rvert$	-2.443E-03	0.0001	0.6122
$\bar{u}_p\bar{w}$	4.884E-04	0.0000	0.6183
$\sin(\phi)\mu_z^2$	2.513E-04	0.0000	0.6246
$\bar{\omega}_{tot}\bar{p}$	5.063E-02	0.0035	0.6297
$\bar{\omega}_{tot}\bar{u}_p^3\lvert\bar{u}_q\rvert$	2.446E-03	0.0004	0.6348
$\lvert\bar{u}_r\rvert\lvert\bar{u}\rvert\bar{w}$	2.638E-04	0.0001	0.6376
$\bar{u}_p\bar{w}\bar{v}_{in}$	-3.335E-06	0.0000	0.6403
$\sin(\phi)\bar{u}_p^2\lvert\bar{u}_r\rvert$	-1.037E-03	0.0002	0.6442

Table 19 Identified polynomial model of C_m for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	Coefficient	Covariance	$\mathbf{R^2}$
$bias$	-4.252E-05	0.0000	0.0000
$\bar{q}$	1.048E-02	0.0006	0.0001
$\bar{u}_q$	5.862E-03	0.0000	0.2811
$\bar{\omega}_{tot}\bar{u}_q$	-2.097E-02	0.0000	0.5413
$\bar{\omega}_{tot}\lvert\bar{u}_r\rvert$	6.724E-04	0.0000	0.6019
$\cos(\theta)\bar{u}_q$	-2.905E-03	0.0000	0.6262
$\cos(\theta)\mu_{v_{in}}\bar{\omega}_{tot}^3$	8.747E-03	0.0002	0.6549
$\sin(\theta)\bar{u}_q^4$	-1.521E-04	0.0000	0.6710
$\bar{\omega}_{tot}\lvert\bar{p}\rvert\lvert\bar{r}\rvert$	1.874E-01	0.0091	0.6838
$\bar{\omega}_{tot}\bar{u}$	4.474E-04	0.0000	0.6935
$\bar{\omega}_{tot}\lvert\bar{p}\rvert^3\bar{q}$	7.053E+01	3.3925	0.7020
$\bar{q}\mu_{v_{in}}$	-3.122E-03	0.0004	0.7079
$\bar{q}\lvert\bar{v}\rvert^2$	-1.358E-02	0.0020	0.7122

Table 20 Identified polynomial model of C_n for the MetalMeetle (Indoors). The order of the tabulated regressors indicates their selection order, with the fixed regressors appearing first in grey rows. Along with each of the selected regressors, their associated coefficient values, and corresponding coefficient variances as a percentage of this value are shown. The coefficient of determination, R^2, describes the fit of the model upon the addition of each regressor during training.

Regressor	**Coefficient**	**Covariance**	**R^2**
$bias$	2.086E-05	0.0000	0.0000
$\bar{r}$	-5.301E-03	0.0001	0.0005
$\bar{u}_r$	3.283E-04	0.0000	0.2960
$\bar{\omega}_{tot}\bar{u}_r^3$	-5.989E-04	0.0000	0.3373
$\mu_{v_{in}}\lvert\bar{p}\rvert\bar{r}^3$	-1.327E+00	0.2168	0.3845
$\bar{\omega}_{tot}\bar{u}_r$	-2.784E-03	0.0001	0.4084
$\bar{\omega}_{tot}\lvert\bar{p}\rvert^3\lvert\bar{q}\rvert$	-2.872E+02	4.2181	0.4238
$\lvert\bar{p}\rvert^2\lvert\bar{q}\rvert$	5.414E+00	0.1113	0.4563
$\cos(\psi)\bar{u}_r$	-1.496E-04	0.0000	0.4701
$\sin(\psi)\bar{u}_r$	-1.318E-04	0.0000	0.4820
$\bar{v}_{in}\lvert\bar{q}\rvert\bar{r}^3$	-1.257E+01	0.6038	0.4928
$\bar{v}_{in}\bar{r}^2$	1.665E-03	0.0001	0.5031
$\sin(\phi)\lvert\mu_x\rvert^2\mu_z$	1.192E-03	0.0001	0.5154

Part III

Prelimenary report

Delft University of Technology

Literature Review: High speed system identification of (un)damaged quadrotors

Supervisor:
Dr. ir. Coen C. de Visser

Author:
Jasper J. van Beers 4479068

Abstract

Quadrotors are an increasingly popular platform for various research fields and a plethora of real-life applications. Recent literature has seen the advancement of quadrotor models, controllers, and safety. Given the limitations of simple first principle models of the quadrotor, considerable interest has been geared towards the development of robust and compelling controllers that push the boundaries of the flight envelope. However, less attention has been directed towards developing better models. Subsequently, much remains unclear about the dynamics behind the quadrotor, particularly in the high-speed regime. Many controllers depend, in some way, on accurate quadrotor models suggesting that such models can only extend the capabilities of the quadrotor. This document aims to highlight notable achievements in quadrotor research over the previous decade and describe the current state-of-the-art. Through this, current gaps in literature, and potential steps forward, may be discussed. Most notably, current research is frequently conducted indoors where external motion capturing systems can facilitate state estimation and where the influence of external disturbances is minimal. It is evident that there is a need for an efficient system identification routine for an undamaged quadrotor suitable for high-speed outdoor applications and aggressive manoeuvres reliant only on on-board sensors.

February 23, 2021

3 Introduction

Research into multirotor aerial vehicles (MAVs) has seen considerable interest in recent years due to the increasing autonomy and versatility of such vehicles. MAVs are subsequently used, both indoors and outdoors, for applications such as videography, surveying and construction, maintenance, agriculture and more. This interest has also culminated in numerous advancements in MAV technology such as developments in fault tolerant controllers [1, 2, 3, 4, 5], the extension of models to the high speed regime [6, 7, 8], to enable aggressive manoeuvres [9, 10, 11], and towards determining the Safe Flight Envelope of MAVs [12].

Among the family of MAVs, the quadrotor in particular remains a popular choice given its simplicity, energy efficiency [13], and low cost of maintenance [14, 15]. As with many MAVs, the versatility of quadrotors is facilitated by their ability to hover, Vertical Take-Off and Landing capabilities [15], and their desirable manoeuvrability characteristics. For instance, quadrotors have been shown to perform aggressive manoeuvres such as flying through narrow openings at various angles [9]. However, the commonly used first principle model of the quadrotor ignores the complex aerodynamic effects which are ubiquitous during aggressive manoeuvres. In fact, this model is inadequate for the majority of the flight envelope, including under high speed conditions [1]. Peculiarly, quadrotors are still able to operate in the high-speed regime and conduct aggressive manoeuvres. This accomplishment is facilitated by high sensor update rates and the use of robust controllers.

It is no surprise, then, that a considerable body of quadrotor literature is invested in developing more compelling and robust quadrotor controllers in spite of poor model fidelities. To this end, sensor-based approaches have recently been employed to reduce the model dependence of the controllers. Smeur et al. [11] demonstrate the superior performance characteristics of an incremental nonlinear dynamic inversion (INDI) controller over the conventional PID controller. However, INDI still requires knowledge on the control effectiveness, and therefore still depends on an appropriate model of the quadrotor. Consequently, control of the quadrotor can also be maintained, and even further improved, through a higher fidelity model.

Indeed, significant discrepancies between models and reality are detrimental to quadrotor performance. Molenkamp et al. [10] found that a significant mismatch between the simulated and true quadrotor model often lead to crashes during aggressive manoeuvring. Existing analytical models are limited in their capacity to adequately describe the complex aerodynamic phenomena prevalent in much of the quadrotor's flight envelope. Therefore, much literature often sees quadrotor models obtained directly from measurement data through system identification techniques. The appeal of such approaches is that the unknown effects are implicitly accounted for in the measurement data. For example, gray-box quadrotor models have recently been identified by Sun et al. [6, 7] which were shown to match the performance of the first principle quadrotor model in near-hover conditions and outperform it in high-speed conditions.

However, such data-driven approaches may be impractical since sufficient data needs to be collected to develop a reliable model. Moreover, while the system identification techniques employed may be applicable to a multitude of quadrotors, the resultant model derived from data is specific to a given quadrotor and configuration (e.g. with or without bumpers). Furthermore, high fidelity quadrotor models, including those of [6, 7], currently require state information obtained from an external motion capturing system due to limitations with the on-board sensors [1, 2, 3]. Hence, while such research is directed towards high-speed outdoor applications, the proposed implementations require adequate on-board state estimation and processing before they see practical use outdoors. Recent literature on such on-board state estimation, such as [16, 17], show promise and may facilitate this transition to outdoor research.

In tandem with this transition, the safety and reliability of the quadrotor platform should be ensured. The simplicity of the quadrotor is, in part, achieved by sacrificing redundancy in the rotors. Coupled with the fact that a quadrotor is a trivially under-actuated system, actuator failure is a considerable concern to the safety of the quadrotor. If nothing is done to account for such failures, the quadrotor may have to abort its mission or, in the worst case scenario, may result in an accident [5]. In light of this lack-of-redundancy, much research has also examined fault tolerant control (FTC) schemes for quadrotors in order to mitigate the consequences of partial [18, 19, 20] and complete [21, 22] actuator failures. Partial failures may be accommodated through explicit modelling [19] or treated as uncertainties to be handled by the controller [20]. Conversely, complete rotor losses necessitate a modification of the control strategy whereby yaw control is typically surrendered [22]. Recent indoor flight tests conducted by Sun et al. [1, 2, 5] have demonstrated the feasibility of such FTC approaches in both the single and diagonal double rotor failure scenarios.

It is evident that high fidelity models of the quadrotor are valuable for extending their capabilities and to facilitate their accepted use in outdoor applications. Much is still unknown about the dynamics behind the quadrotor, especially outside the low-speed domain. Therefore, this document is concerned with presenting outcomes in recent quadrotor literature and is structured as follows: chapter 4 provides a brief history of the quadrotor highlighting notable advancements and concludes with a simple first principle quadrotor model, used extensively in literature. Some analytical extensions to this simple quadrotor model, aimed at improving model accuracy, are presented in chapter 5. Recent literature has been concerned with the fault tolerant control of the quadrotor. Therefore, chapter 6 highlights efforts made in literature to accommodate the partial and full failure of the rotors. Given the complexity of the quadrotor dynamics, analytical approaches to modelling are ill-equiped to describe the quadrotor under conditions of fault or when operating along the boundaries of its flight envelope. Instead, data-driven system identification approaches are employed, some of which are discussed in chapter 7. In order to facilitate such techniques, estimations on the quadrotor state are necessary. Some state estimation approaches are therefore summarized in chapter 8. This document concludes with a research proposal, in chapter 9, aimed at addressing some of the gaps in literature discussed in the preceding chapters.

4 An overview of the quadrotor

The quadrotor remains a favorite among the family of micro aerial vehicles (MAVS) due to its simplicity and flight capabilities. While the definition of what constitutes a quadrotor may be up for interpretation, a general definition of a 'quadrotor' may be considered as follows:

Definition 4.0.1

A **quadrotor** is an aerial vehicle which makes use of four propellers, typically in a cross-like configuration, to provide lift, thrust, and attitude control facilitating the motion of the vehicle [23].

Note that in this definition, the quadrotor does not necessarily have to be a MAV. As outlined in section 4.1 on the history of the quadrotor, in contrast to the colloquial quadrotors of today, many of the earlier iterations were in fact large and manned vehicles. As such, to fully describe the history of the quadrotor, a more broad definition is used in this chapter. Despite this, the quadrotor treated in subsequent chapters is primarily the MAV-variant seeing as much of the contemporary research is conducted on this subclass of the quadrotor. To facilitate these discussions, a simple model of the (MAV) quadrotor, used extensively as a benchmark in literature, is summarized in section 4.2.

4.1 A brief history of the quadrotor

Widely regarded as the first successful demonstration of flight with a quadrotor is the Brèguet-Richet *Gyroplane No. 1* in 1907 [23, 24, 25, 26]. Unlike contemporary quadrotors, the the only controllable input to the *Gyroplane No. 1* was the rotor speed, all four of which were connected via a single engine [25]. As such, the *Gyroplane No. 1* was difficult to control, unstable and only capable of vertical flight. While reports on the exact altitudes reached by the first quadrotor differ, it is believed that the *Gyroplane No. 1* achieved an altitude between 2 [24] and 5 [25] feet (0.6 and 1.5 meters respectively), even if only for a brief time. Regardless, the *Gyroplane No. 1* was able to demonstrate that vertical flight was indeed possible and, while impractical as a quadrotor [26], this design inspired subsequent quadrotor research.

It was not until the early 1920s when more flight capable quadrotors were constructed [24, 26, 27]. For instance, Étienne Oehminchen's *Oehmichen No. 2* made use of blade warping to modify the angle of attack of the two-bladed rotors, enabling improved control of the vehicle over the *Gyroplane No. 1* [26]. Two additional propellers were fixed laterally and placed at the nose of the *Oehmichen No. 2* to facilitate yaw control, hence some consider this vehicle a

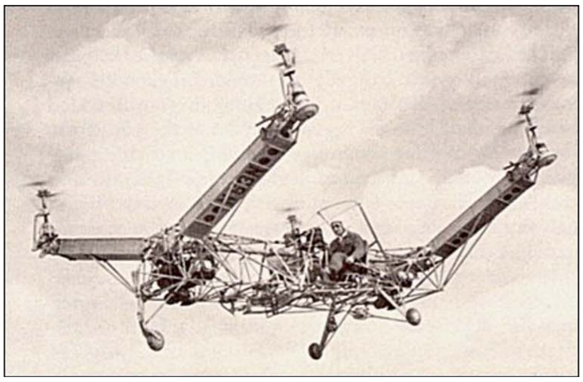


Figure 4.1: The Convertawings Model A (1956) [26]

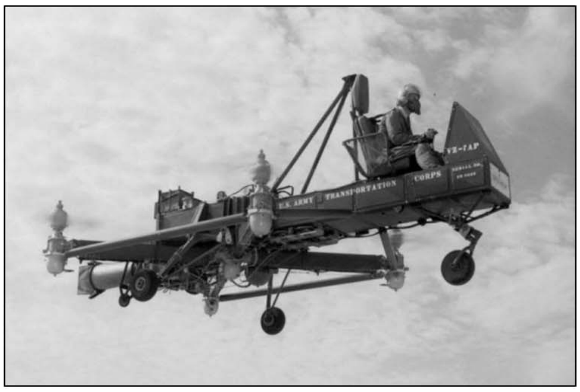


Figure 4.2: The Curtiss-Wright VZ-7 (1958) [26]

helicopter-quadrotor hybrid [26]. Other quadrotors were also developed around this time, but due to limitations in the current technology, many of these quadrotor endeavours failed and the popularity of such vehicles dwindled [24, 26].

Further innovations in quadrotor designs did not materialize until the mid 1950s [24, 26]. The primary innovation of these quadrotor designs was the implementation of differential thrust which enabled improved control and forward flight of the quadrotors [24, 26]. Two pioneering designs demonstrated the feasibility of forward flight, namely the *Convertawings Model A* (1956) and *Curtiss-Wright VZ-7* (1958) [26]. The *Convertawings Model A* made use of two engines to control the four propellers [26]. Thus, the attitude of this quadrotor was controlled through variable pitch of the blades and thrust of the rotors [28]. Perhaps more representative of modern quadrotors, the *Curtiss-Wright VZ-7* made use of a single motor per rotor and was thereby able to control both forward flight and attitude through differential thrust [29]. However, a lack of interest lead to the eventual failure of these quadrotors [26] and the stagnation of innovation towards quadrotor technology.

Interest in quadrotors was once again rekindled in the early 1990s whereby advancements in micro electro-mechanical systems (MEMS) allowed for the development of micro controllers [23, 24, 27]. Consequently, much of the low-level stability control could now be done automatically on-board the quadrotor itself, using various sensors also attached to the vehicle [27]. Along with these innovations came the transition from large manned quadrotors to comparatively small and mostly autonomous vehicles. The small size, autonomy, low-cost and accessibility of these MEMS-equipped quadrotors made them an attractive asset for both civilian and military use [11, 23]. Moreover, given the rapid pace of advancements in other domains, such as those of mobile and smartphone technology, quadrotors are only growing in popularity in numerous industries such as construction, agriculture, videography, and sustainability [5, 11]. To facilitate this growing interest, much research has been - and continues to be - focused on quadrotor technology in both the commercial and academic sectors [23, 24, 27, 26].

One of the earliest commercial quadrotors was the Draganflyer series, which has seen extensive use in research by virtue of its distinguished performance [24]. Many other companies and institutions have since either developed their own quadrotor platforms, such as the Standford's STARMAC I & II [27, 15, 30] (see fig. 4.3), or modified existing commercial quadrotors, such as the Parrot Bebop 1 shown in fig. 4.4 (as is done by Li et al. in [8]). Moreover, the emergence of open-source and customizeable autopilot software, such as Paparazzi [31], has made research into MAVS more accessible. The choice of either building a quadrotor or using a commercial one largely depends on the purpose of research and requirements thereof. In control-based research, using custom software on commercial drones or off-the-shelf components is more common since applicability to common and existing quadrotors is often a motivating factor of research.

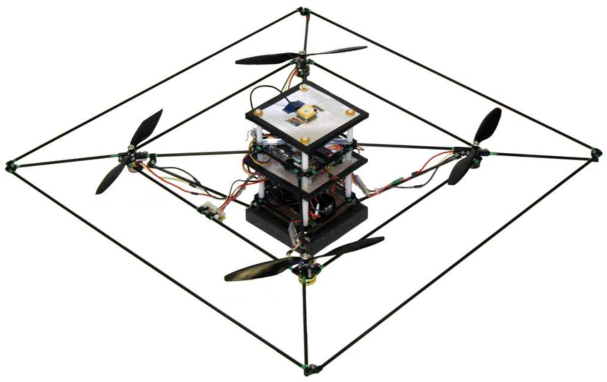


Figure 4.3: The STARMAC II quadrotor platform developed by Standford University [15]

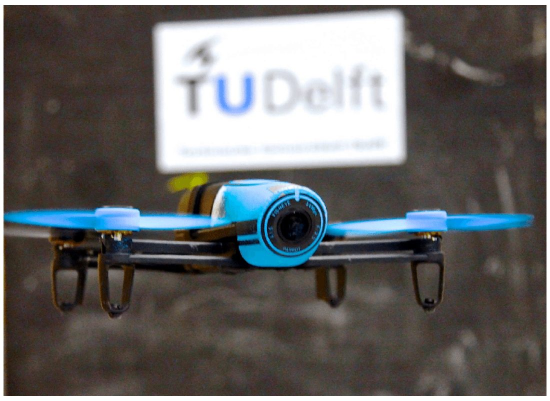


Figure 4.4: The (modified) Parrot Bebop 1 commercial quadrotor [8]

4.2 A first principle model of the quadrotor

In this section a simple mathematical model of the quadrotor is defined, based on first principles. To describe the position and orientation of the quadrotor, two reference systems are first constructed in section 4.2.1. Based on these reference systems, the simple quadrotor model is derived and presented in section 4.2.2. While this model is specific to the established reference system and quadrotor, the first principle approach used may be generalized to other multirotors.

4.2.1 Reference frames

In order to describe the motion (i.e. position and orientation) of the quadrotor in space, two reference frames are used. The inertial reference frame, denoted by $\{E\} = \{O_E, x_E, y_E, z_E\}$, is fixed to a point on the ground and is oriented using the north-east-down (NED) configuration as shown in fig. 4.5. The position of the drone in space is usually expressed with respect to this inertial reference frame.

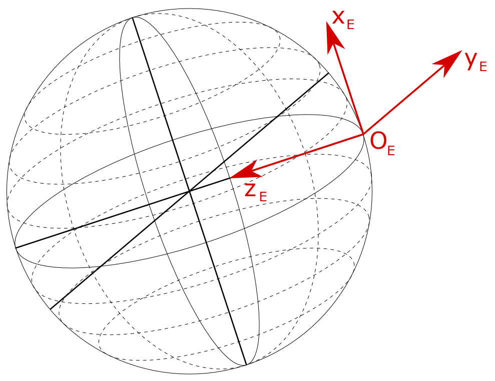


Figure 4.5: Illustration of the inertial reference frame, $\{E\}$, following the North-East-Down (NED) configuration with origin at O_E.

To describe the orientation of the drone with respect to the inertial frame, the body reference frame, $\{B\} = \{O_B, x_B, y_B, z_B\}$, is defined. As the name implies, this reference frame is fixed to the quadrotor with origin, O_B, at the center of gravity of the drone, x_B pointing forwards, y_B to the right, and z_B aligned with gravity when the drone is hovering as illustrated in fig. 4.6. In the rest of this document, the superscript B in A^B indicates that the entity A is expressed in the body frame, $\{B\}$. No superscript means that A is expressed in the inertial frame, $\{E\}$, unless stated otherwise.

The (orientation of the) body frame can be related to the inertial frame through the euler angles: ϕ (roll), θ (pitch), and ψ (yaw). These angles are illustrated in fig. 4.7 wherein the faded blue quadrotor denotes the orientation which is aligned with $\{E\}$. The rotation matrix about each axis is given by eq. 4.1 wherein $c(\cdot)$ and $s(\cdot)$ denote the cosine and sine operators respectively (i.e. $c(\cdot) = cos(\cdot)$ and $s(\cdot) = sin(\cdot)$).

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi) & c(\phi) \end{bmatrix}, \quad R_y(\theta) = \begin{bmatrix} c(\theta) & 0 & s(\theta) \\ 0 & 1 & 0 \\ -s(\theta) & 0 & c(\phi) \end{bmatrix}, \quad R_z(\psi) = \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Rotations about multiple axes can be given by a product of the appropriate rotation matrices defined in eq. 4.1. For example, the rotation matrix from the body frame to the inertial frame can be obtained by first rotating about the x-axis, then y-axis and finally the z-axis. Note that this order is reversed in the subscript of the rotation matrix R (i.e. this rotation is shown as

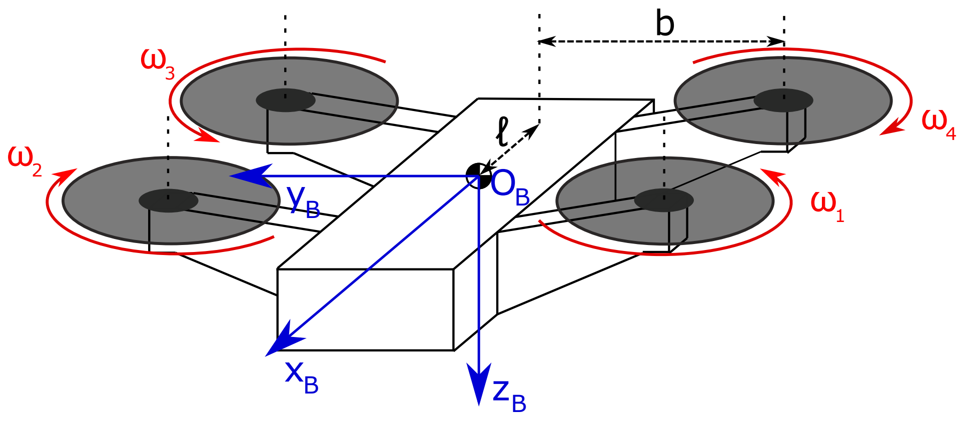


Figure 4.6: Illustration of the generic quadrotor in the breaststroke configuration, whereby rotor one is rotating counterclockwise, denoted by ω_1. Depicted is the body reference frame, B, with origin, O_B, at the quadrotor's center of gravity (c.g.). Also shown are geometric parameters b and ℓ, which respectively represent the y_B and x_B distances from the rotor centers of rotation to the c.g. of the quadrotor.

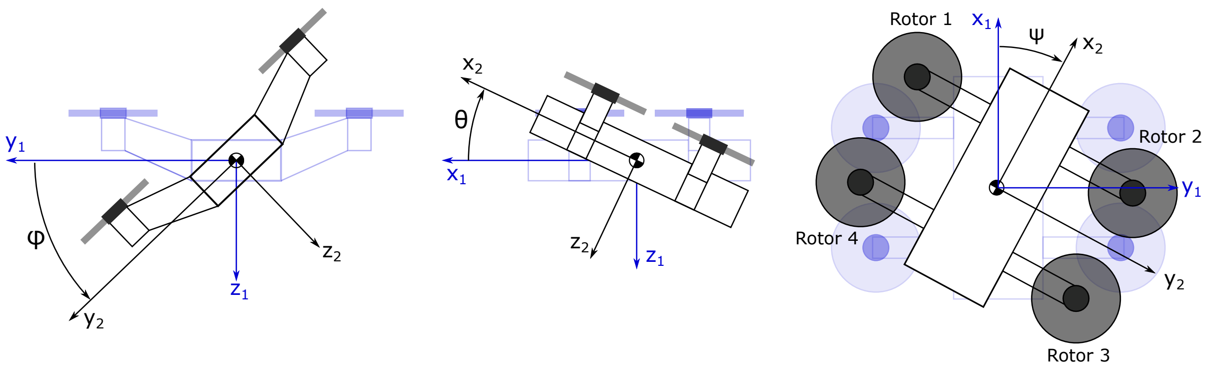


Figure 4.7: Definitions of the Euler Angles: ϕ (Roll), θ (Pitch), and ψ (Yaw)

R_{zyx}; meaning that the z-rotation is performed after the y-rotation which is performed after the x-rotation). Equation 4.2 denotes the resultant rotation matrix from body to inertial reference frame, R_{EB}.

$$R_{EB} = R_{zyx}(\phi,\theta,\psi) = R_x(\phi)R_y(\theta)R_z(\psi)$$

$$R_{EB} = \begin{bmatrix} c(\theta)c(\psi) & s(\phi)s(\theta)c(\psi)-c(\phi)s(\psi) & c(\phi)s(\theta)c(\psi)+s(\phi)s(\psi) \\ c(\theta)s(\psi) & s(\phi)s(\theta)s(\psi)+c(\phi)c(\psi) & c(\phi)s(\theta)s(\psi)-s(\phi)c(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{4.2}$$

Intuitively, the inertial frame can be aligned with the body frame by rotating through the negatives of the euler angles in the reverse order. Therefore, the resultant rotation matrix from inertial frame to body frame, R_{BE}, is given by eq. 4.3.

$$R_{BE} = R_{xyz}(\phi,\theta,\psi) = R_z(-\psi)R_y(-\theta)R_x(-\phi)$$

$$R_{BE} = \begin{bmatrix} c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\ s(\phi)s(\theta)c(\psi)-c(\phi)s(\psi) & s(\phi)s(\theta)s(\psi)+c(\phi)c(\psi) & s(\phi)c(\theta) \\ c(\phi)s(\theta)c(\psi)+s(\phi)s(\psi) & c(\phi)s(\theta)s(\psi)-s(\phi)c(\psi) & c(\phi)c(\theta) \end{bmatrix} \tag{4.3}$$

4.2.2 Quadrotor dynamics

With the reference frames defined, a simple model of the quadrotor may be constructed. Let the position of the quadrotor in the inertial frame be given by

$$\boldsymbol{\xi} = \begin{bmatrix} x & y & z \end{bmatrix}^T$$

The velocity of the quadrotor can be obtained by taking the derivative of the position as illustrated by eq. 4.4 with the velocity vector defined as $\mathbf{V} = \begin{bmatrix} u & v & w \end{bmatrix}^T$.

$$\mathbf{V} = \dot{\boldsymbol{\xi}} \tag{4.4}$$

Using this relation, along with the assumption that the quadrotor is a rigid body, a simple model of the quadrotor can be derived by following Newton's laws of motion. The resultant force and moment equations are given by eq. 4.5 and eq. 4.6 respectively.

$$m\dot{\mathbf{V}} = m\mathbf{g} + R_{EB}\mathbf{F}^B \tag{4.5}$$

$$\mathbf{I_v}\dot{\boldsymbol{\Omega}}^B + \boldsymbol{\Omega}^B \times \mathbf{I_v}\boldsymbol{\Omega}^B = \mathbf{M}^B \tag{4.6}$$

In these equations, m denotes the mass of the quadrotor and $\mathbf{I_v}$ gives its moment of inertia. The gravity vector along z_E (refer to fig. 4.5) is given by $\mathbf{g}$. The resultant force acting on the quadrotor, $\mathbf{F}^B$, is a sum of the control forces exerted by the rotors and the aerodynamic forces acting on the quadrotor. Since these forces act on the quadrotor itself, they need to be transformed into the inertial reference frame, hence, R_{EB} represents the rotation matrix from the body frame, $\{B\}$, to the inertial frame, $\{E\}$. Consequently, eq. 4.5 is defined in the inertial frame.

The rotational rates of the quadrotor are given by $\boldsymbol{\Omega} = \begin{bmatrix} p & q & r \end{bmatrix}^T$ and are expressed in the body frame. As such, the total moment acting on the quadrotor, $\mathbf{M}^B$, is also defined in the body frame and is comprised of the control moments, aerodynamic moments, and gyroscopic moments (due to the rotation of the rotors).

Attentive readers may have already noted that eq. 4.5 and eq. 4.6 are defined in different reference frames. Given that the body frame may be rotating with respect to the inertial frame, the velocity in the inertial frame can related to the velocity in the body frame through eq. 4.7. It can be verified that when there are no rotational velocities acting on the body (i.e. $\boldsymbol{\Omega} = \mathbf{0}$), the velocity in the inertial frame is equivalent to that in the body frame.

$$\dot{\mathbf{V}} = \dot{\mathbf{V}}^B + \boldsymbol{\Omega}^B \times \mathbf{V}^B \tag{4.7}$$

Therefore, using eq. 4.7 and expressing the inertial terms with respect to the body frame, eq. 4.5 can be re-written in $\{B\}$ through eq. 4.8

$$m\left(\dot{\mathbf{V}}^B + \boldsymbol{\Omega}^B \times \mathbf{V}^B\right) = R_{BE}m\mathbf{g} + \mathbf{F}^B \tag{4.8}$$

Depending on the complexity of the quadrotor model, different forces and moments may be neglected when considering $\mathbf{F}^B$ and $\mathbf{M}^B$ respectively. However, to control the quadrotor, it is necessary to model the control forces and moments induced by the rotors. Therefore, let

$$\mathbf{F}^B = \mathbf{F}_c^B + \Delta_{\mathbf{F}^B} \tag{4.9}$$

$$\mathbf{M}^B = \mathbf{M}_c^B + \Delta_{\mathbf{M}^B} \tag{4.10}$$

where $\mathbf{F}_c^B$ and $\mathbf{M}_c^B$ denote the resultant control force and moments generated by the rotors respectively. Similarly, $\Delta_{\mathbf{F}^B}$ and $\Delta_{\mathbf{M}^B}$ represent the other (e.g. aerodynamic) resultant forces and moments which may also act on the quadrotor.

In a conventional quadrotor, the control forces and moments can be manipulated by controlling the thrust of individual rotors. A simple model of the total thrust generated by the rotors is given by eq. 4.11 where ω_i denotes the angular velocity of the i^{th} rotor[†] and κ_0 is a constant for the quadrotor which represents properties of the rotor and air density [6, 32, 33]. Note that κ_0 is typically identified through measurements of the hovering quadrotor in windless conditions where the influences of $\Delta_{\mathbf{F}^B}$ are negligible, i.e., aerodynamic effects are negligible [33]. Hence, $\mathbf{F}^B = \mathbf{F}_c^B$ when hovering.

$$T = \kappa_0 \sum \omega_i^2 \tag{4.11}$$

Since the resultant thrust, T, acts in the negative z_B direction, the control forces exerted on the conventional quadrotor can be summarized through eq. 4.12.

$$\mathbf{F}_c^B = \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\kappa_0 \sum \omega_i^2 \end{bmatrix} \tag{4.12}$$

The control moment exerted on the quadrotor depends on the geometry of the quadrotor. Namely, the arm lengths from the rotors to the center of gravity. Following fig. 4.6, these arm lengths are b (along the y_B axis) and ℓ (along the x_B axis). The (rolling) moment about x_B is generated through differential thrust of rotors 1 & 4 and rotors 2 & 3 and can be given by

$$M_x^B = b\kappa_0 \left(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2\right)$$

Likewise, the (pitching) moment about y_B is governed by rotors 1 & 2 and rotors 3 & 4 and can be expressed through

$$M_y^B = \ell\kappa_0 \left(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2\right)$$

Unlike the moment about x_B or y_B, the moment about z_B (yaw axis) is controlled by the moments produced by the rotation of an individual rotor and therefore depends on the configuration of the quadrotor [9]. Figure 4.6, illustrates the 'breaststroke' configuration: wherein rotor 1, ω_1, rotates counterclockwise. An alternative configuration is known as the 'bear-hug' configuration which has rotor 1 rotating clockwise. In fig. 4.6, the rotations of rotors 1 and 3 produce a negative moment while those of rotors 2 and 4 produce a positive moment. Consequently, the moment about z_B is given by

$$M_z^B = \tau_0 \left(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2\right)$$

where τ_0 is a constant called the rotor torque coefficient and, like κ_0, may likewise be identified from measurements taken during hover under no-wind conditions. Therefore, the control moment, $\mathbf{M}_c^B$, is given by eq. 4.13.

$$\mathbf{M}_c^B = \begin{bmatrix} b\kappa_0 & -b\kappa_0 & -b\kappa_0 & b\kappa_0 \\ \ell\kappa_0 & \ell\kappa_0 & -\ell\kappa_0 & -\ell\kappa_0 \\ -\tau_0 & \tau_0 & -\tau_0 & \tau_0 \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{4.13}$$

[†]It is assumed that the rotors are perfectly flush with the x_B-y_B plane. Therefore, the angular rate of the rotor is only about the z_B-axis. Note that this is a simplification and, in reality, imperfections in the manufacturing and assembly will cause slight deviations. However, the majority of the angular rate will be around the z_B-axis for conventional designs and thus the out-of-axis thrust produced is negligible.

Some literature, such as [6], also model the gyroscopic effects and rotor inertia in the base model. These effects will be neglected for the base model presented in this section and will instead be treated in the subsequent chapter. In the absence of aerodynamic effects, and neglecting the gyroscopic and inertial effects of the rotor, the resultant moment acting on the quadrotor simplifies to $\mathbf{M}^B = \mathbf{M}_c^B$.

With this, a simple dynamic model of the quadrotor is given by eq. 4.6 and eq. 4.8 with forces and moments described by eq. 4.12 and eq. 4.13 respectively. However, this simplified model is only valid in the near-hover regime. Consequently, there is a considerable amount of literature centered around developing a more accurate model of the quadrotor. Some of the most commonly seen, primarily analytical, extensions to the simplified quadrotor model are therefore treated in chapter 5.

5 Extensions to the simple quadrotor model

While the simple quadrotor model defined in chapter 4 (section 4.2) can describe the quadrotor, and is often used as a basis for several quadrotor models, it is only valid for a limited region of the flight envelope. In fact, as will be highlighted in this chapter, this model is only valid under near-hover conditions (i.e. $< 2ms^{-1}$ [7]). Therefore, the simple model inadequately describes the quadrotor in the majority of the flight envelope, including simple translational flight. These model inaccuracies only grow when the quadrotor operates at high-speeds and/or performs aggressive manoeuvres. For example, Sun et al. [7] found that the performance of such hovering models completely deteriorates above $5ms^{-2}$. Moreover, the simple model does not consider external disturbances, which may be quite prominent in outdoor environments (e.g. the presence of wind).

Therefore, to account for some of these effects, various extensions to the simple model are proposed in literature. Some of the most common and relevant modifications are discussed in this chapter. Note that most of these extensions are derived analytically from physical principles. Of course, data-driven approaches also exist and see widespread use in literature (e.g. [6, 7, 8, 34, 35]). However, such approaches are treated in chapter 7.

Some literature sources, such as Sun et al. [6], consider the effects of rotor inertia's and the gyroscopic moments induced by these motors to be non-negligible even in hover. Hence, section 5.1 discusses these effects. The presence of wind is something that cannot be ignored for MAVs operating outdoors and is therefore treated in section 5.2. Related to the effects of wind are aerodynamic forces and moments which act on the quadrotor during both hovering and translational flight. Namely, the influence of thrust variance and blade flapping are presented in section 5.3.

5.1 Rotor inertia and gyroscopic moment

The rotation of the quadrotor propellers induces a gyroscopic moment on the quadrotor. This moment can be described by eq. 5.1 where the index, i, denotes the i^{th} propeller. Here, $\mathbf{I}_r$ gives the moment of inertia of the propeller, $\mathbf{\Omega}^B$ describes the rotation rate of the quadrotor, $\mathbf{M}_{r_i}$ represents the moment induced by propeller i and $\boldsymbol{\omega}_i$ denotes the rotation rate vector of the i^{th} propeller [11].

$$\mathbf{I}_r\dot{\boldsymbol{\omega}}_i^B + \mathbf{\Omega}^B \times \mathbf{I}_r\boldsymbol{\omega}_i^B = \mathbf{M}_{r_i}^B \tag{5.1}$$

Decomposing eq. 5.1 into the individual components along the body axis results in

$$\mathbf{M}_{r_i}^B = \begin{bmatrix} \mathbf{M}_{r_{i_x}}^B \\ \mathbf{M}_{r_{i_y}}^B \\ \mathbf{M}_{r_{i_z}}^B \end{bmatrix} = \begin{bmatrix} I_{r_{xx}}\dot{\omega}_{i_x} - I_{r_{yy}}\Omega_z\omega_{i_y} - I_{r_{xy}}\Omega_z\omega_{i_x} + I_{r_{zz}}\Omega_y\omega_{i_z} \\ I_{r_{yy}}\dot{\omega}_{i_y} + I_{r_{xx}}\Omega_z\omega_{i_x} + I_{r_{xy}}\Omega_z\omega_{i_y} - I_{r_{zz}}\Omega_x\omega_{i_z} \\ I_{r_{zz}}\dot{\omega}_{i_z} - I_{r_{xx}}\Omega_y\omega_{i_x} - I_{r_{xy}}\Omega_y\omega_{i_y} + I_{r_{yy}}\Omega_x\omega_{i_y} + I_{r_{xy}}\Omega_x\omega_{i_x} \end{bmatrix} \tag{5.2}$$

In eq. 5.2, $I_{r_{xx}}$, $I_{r_{yy}}$, and $I_{r_{zz}}$ denote the moments of inertia about the x_B, y_B, and z_B axes respectively. $I_{r_{xy}}$ is the product of inertia of the x_B and y_B axes. The rotational velocities of the individual rotors, i, about the x_B, y_B, and z_B axes are given by ω_{i_x}, ω_{i_y}, and ω_{i_z} respectively. Similarly, $\mathbf{\Omega} = [\Omega_x, \Omega_y, \Omega_z]^T = [p, q, r]^T$.

Note that the term $I_{r_{xz}}$ and $I_{r_{yz}}$ are absent (i.e. $I_{r_{xz}} = I_{r_{yz}} = 0$) since it is assumed that the rotors are flush with the x_B-y_B-plane (as is done in chapter 4). For unique quadrotors,

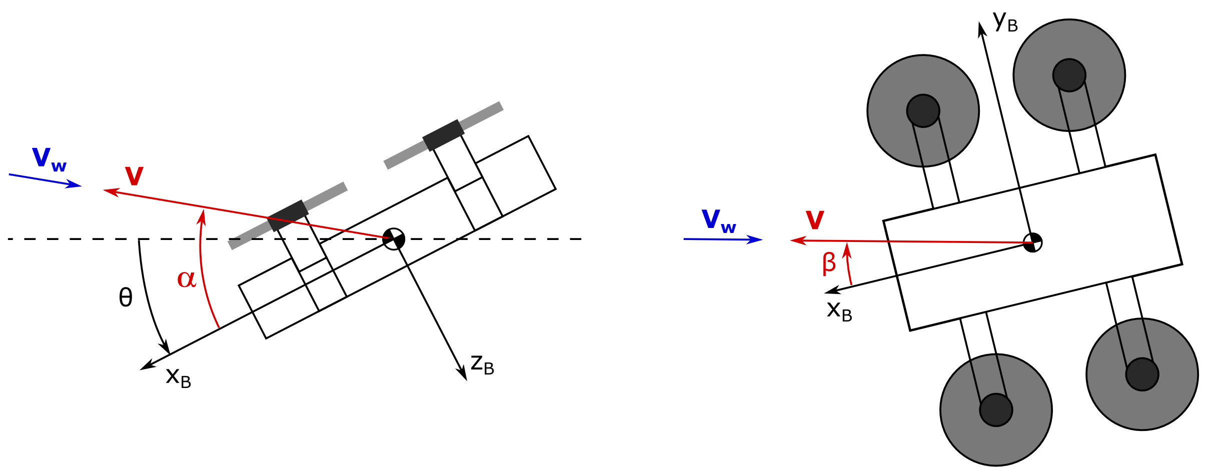


Figure 5.1: Illustration of the angle of attack, α, (left) and angle of sideslip, β, (right) with respect to the quadrotor body frame.

wherein such propeller symmetry does not hold or the propellers are not flat in the z-axis, then the appropriate product of inertia terms should be included in eq. 5.2.

Since the propellers rotate about the z_B axis, their rotational velocities are much greater than along either the x_B or y_B axes (i.e. $\omega_{i_z} \gg \omega_{i_x}$ and $\omega_{i_z} \gg \omega_{i_y}$ and $\dot{\omega}_{i_x}$ and $\dot{\omega}_{i_y}$ are negligible) [11]. Moreover, the inertia of the quadrotor is much larger than that of the propellers (i.e. $\mathbf{I}_r \ll \mathbf{I}_v$). Therefore, the only significant terms in eq. 5.2 are those with large ω_i, which are terms containing ω_{i_z} or its derivative. Equation 5.2 can therefore be reduced to

$$\mathbf{M}^B_{r_i} \approx \begin{bmatrix} I_{r_{zz}}\Omega_y\omega_{i_z} \\ -I_{r_{zz}}\Omega_x\omega_{i_z} \\ I_{r_{zz}}\dot{\omega}_{i_z} \end{bmatrix} \tag{5.3}$$

5.2 Presence of wind

Since MAV research has only recently seen major growth, much of the quadrotor research has been conducted indoors wherein the influence of wind is negligible. However, a multitude of quadrotor applications involve outdoor operations wherein the influence of external disturbances cannot be omitted. In this section, only a simple representation of wind is given. It is noteworthy to highlight that modelling wind on its own is a challenging task, given its stochastic nature. Such models are out of the scope of this paper. Hence, in this section, it is assumed that the instantaneous wind vector is known in $\{E\}$.

In chapter 4 (specifically, section 4.2), the velocity vector $\mathbf{V}$ used represents the *airspeed* of the quadrotor and is therefore influenced by the presence of wind. Moreover, depending on the magnitude of the wind, complex aerodynamic effects may be at play even if the quadrotor appears to be barely moving with respect to the inertial frame. Much of the research into quadrotor dynamics and control is conducted in windless conditions (see, for instance, [5, 8, 9, 17, 23]). While this is perhaps suitable for indoor applications (such as drone racing), ignoring wind is problematic for outdoor applications where such effects are typically non-negligible.

The airspeed of the quadrotor, $\mathbf{V} = \begin{bmatrix} u & v & w \end{bmatrix}^T$ is related to the ground speed, $\mathbf{V_G}$, and relative wind speed, $\mathbf{V_W}$, through eq. 5.4 [7].

$$\mathbf{V} = \mathbf{V_G} - \mathbf{V_W} \tag{5.4}$$

The angle of attack, α, is the angle between the $x_B - y_B$ plane and the relative velocity vector of the aircraft and may be calculated through eq. 5.5 [7]. The angle of sideslip, β, is given by the angle between the $x_B - z_B$ plane and the relative velocity vector which is related to the airspeed through eq. 5.6 [7]. For clarity, the definition of these angles are also depicted in fig. 5.1.

$$\alpha = arcsin\left(\frac{w}{||\mathbf{V}||}\right) \tag{5.5}$$

$$\beta = arcsin\left(\frac{v}{\sqrt{v^2+u^2}}\right) \tag{5.6}$$

Given that α and β are defined with respect to the relative velocity vector, conditions with $\mathbf{V} = \mathbf{0}$ result in singular angles. It is evident from eq. 5.4 that this condition is satisfied when the ground speed equals the wind speed. A special case of this occurs when the drone is hovering under conditions of no wind. Given the prevalence of hovering in quadrotor operations, special care should be given to account for such singularities.

5.3 Aerodynamic effects

Due to their complexity, aerodynamic effects are often neglected in quadrotor models. While this may be suitable for a nominal quadrotor in near-hovering conditions [36], such simplifications are incompatible with the dynamics at higher velocities, during aggressive manoeuvres, or even for damaged quadrotors. In other words, neglecting these aerodynamic effects confines the quadrotor model to a narrow region of the flight envelope thereby restricting its capabilities. Consequently, much literature has been focused on developing accurate aerodynamic models for the quadrotor. Such literature draws heavily from extensive helicopter literature [36] with adjustments made to harmonize with the quadrotor dynamics. In this section, two effects that see frequent attention in literature are discussed: the influence of thrust variation and the influence of blade flapping.

5.3.1 Thrust variation

One of the prominent aerodynamic effects which comes into play at higher velocities is the increased dependence of thrust - and therefore altitude control - on the relative velocity between the rotor and surrounding air, V (see eq. 5.4), and the angle of attack of the rotor disks, α_r [15, 36]. These variables are not taken into account for the simple thrust model presented in chapter 4 (i.e. eq. 4.11) which only depends on the angular rate of the rotors, $\omega_j \quad \forall \quad j \in [1, 4]$.

Air gains additional energy, through an increased velocity, as it passes through the rotor. Let this additional velocity be the induced velocity, v_i. The thrust of the quadrotor, at arbitrary α_r, can be found using momentum theory as shown in eq. 5.7 [33]. Here, α_r is the angle of attack of the rotor disk (i.e. rotor plane, which is assumed to lie in line with the $x_B - y_B$ plane). The mass flow, $\dot{m}$, is a function of the area swept by the rotor blade, A, and the air density ρ.

$$T = 2\rho A v_i \sqrt{V^2 + 2V\sin(\alpha_r)v_i + v_i^2} \tag{5.7}$$

In order to determine the thrust in eq. 5.7, the induced velocity v_i needs to be known. This quantity can be rather difficult to obtain, so another expression relating the thrust and induced velocity is needed to complete the system of equations. Blade element theory can be used to derive another equation for the thrust which also depends on V, α_r, and the rotor blade geometry. Equation 5.8 represents the thrust derived from blade element theory [33, 37]. It is important to emphasize that eq. 5.8 is ill-suited for conditions where the quadrotor is close to horizontal surfaces (e.g. roof or ground) since it neglects the ground effect.

$$T = \frac{\rho abc \sum_{j=1}^{4} \omega_j^2 R^3}{2} \left(\frac{\theta_r}{3} + \frac{V^2 \cos^2(\alpha_r)\theta_r}{2\sum_{j=1}^{4} \omega_j^2 R^2} + \frac{V \sin(\alpha_r) + v_i}{2\sum_{j=1}^{4} \omega_j R} \right) \tag{5.8}$$

In eq. 5.8, a represents the lift curve slope of the rotor, b the number of blades (typically two for a quadrotor), R denotes the radius of the rotor, c the blade chord, and θ_r the pitch of the rotor blade. Quadrotors, and indeed some helicopters, often harbour twisted rotor blades (i.e. the blade pitch varies along the blade span) and chord lengths which may also vary along the blade span. In the underlying helicopter literature [37], it is assumed that the chord, c, is constant. This assumption is subsequently carried over into quadrotor literature by Powers et al. [33]. Powers et al. [33] further simplify the blade geometry by assuming fixed pitch blades, contributing to some of the observed modelling errors. For twisted blades, Johnson [37] proposes two representative values for θ_r depending on the nature of the twist. For linearly twisted blades, θ_r may be represented by the pitch at 75% of the rotor's radius while for ideally twisted blades, θ_r is given by the pitch at the tip of the rotor blade [37].

Regardless of the exact intricacies of the blade geometry itself, the parameters a, b, c, R, and θ_r are constant for a quadrotor [6]. Therefore, these terms can be lumped into constant coefficients and eq. 5.8 can be rewritten as:

$$T = \kappa_1 \sum_{j=1}^{4} \omega_j^2 + \kappa_2 V^2 \cos^2(\alpha_r) + \kappa_3 \left(V \sin(\alpha_r) + v_i\right) \sum_{j=1}^{4} \omega_j \tag{5.9}$$

In the case where the quadrotor is either ascending or descending, i.e. $\alpha_r = \frac{\pi}{2}$, then eq. 5.7 and eq. 5.9 simplifies to eq. 5.10 and eq. 5.11 respectively [33].

$$T = 2\rho A v_i \left|V + v_i\right| \tag{5.10}$$

$$T = \kappa_1 \sum_{j=1}^{4} \omega_j^2 + \kappa_3 \left(V + v_i\right) \sum_{j=1}^{4} \omega_j \tag{5.11}$$

Moreover, in the special case where $V = 0$ (e.g. during hovering conditions), eq. 5.10 can be further simplified and rearranged to obtain the induced velocity during hover, v_h, as given by

$$v_h = \sqrt{\frac{T}{2\rho A}} \tag{5.12}$$

Continuing with the case where $V = 0$ and substituting eq. 5.12 for v_i in eq. 5.11 yields eq. 5.13, relating the thrust to the rotational rate.

$$T = \kappa_1 \sum_{j=1}^{4} \omega_j^2 + \kappa_3 v_h \sum_{j=1}^{4} \omega_j \tag{5.13}$$

In fact, since v_h is also a function of T, eq. 5.13 can be rearranged into quadratic expression in thrust. Solving the resultant quadratic expression leads to a solution given by eq. 5.14 [6, 33]. Note that this solution is analogous to the simplified thrust model presented in section 4.2 and therefore highlights the limitations of the simplified model. Namely, that it is only valid in near-hover conditions (i.e. quadrotor z_B-axis is aligned with inertial frame z_e axis: $\alpha_r \approx \frac{\pi}{2}$; and quadrotor is not moving: $V \approx 0$).

$$T = \kappa_4 \sum_{j=1}^{4} \omega_j^2 \tag{5.14}$$

Outside of hover conditions, the thrust variance effect comes into play [6] and the additional thrust, T_a generated by the rotors can be defined as the actual thrust subtracted by the thrust during hover, T_h, as shown in eq. 5.15.

$$T_a = T - T_h \tag{5.15}$$

Substituting eq. 5.9 and eq. 5.13 (note that here $T = T_h$) into eq. 5.15 yields

$$T_a = \kappa_2 V^2 \cos^2(\alpha_r) + \kappa_3 \left[V \sin(\alpha_r) + (v_i - v_h)\right] \sum_{j=1}^{4} \omega_j \tag{5.16}$$

In eq. 5.16, $(v_i - v_h)$ gives the additional thrust v_a. The induced velocity during hover, v_h, can be obtained experimentally. Therefore, the true induced velocity, v_i, can be computed using eq. 5.17 [15, 36] which can subsequently be used to obtain the thrust variance.

$$v_i = \frac{v_h^2}{\sqrt{(V \cos \alpha_r)^2 + (v_i - V \sin \alpha_r)^2}} \tag{5.17}$$

Note, however, the presence of the induced velocity, v_i, on both sides of eq. 5.17. Therefore, for an arbitrary α_r and relative velocity, V, eq. 5.17 is now a fourth order polynomial with respect to the induced velocity, v_i. Although solving this equation would allow for the computation of the actual thrust, T, it may prove challenging to do so accurately since the equation itself needs to be solved and the correct root needs to be chosen. Solutions to this relation can be obtained experimentally through a rotor test rig as done by Powers et al. [33]. However, in [33], the results found through the experiment were not consistent with data collected during free flight of the quadrotor, even though the model itself was consistent with the test rig results. Powers et al. [33] suspect that this is due to the difference in flow conditions between the rotor test rig and free flight. Regardless, the test rig results indicate that the thrust variation decreases for increasing α_r (i.e. as the z_B axis approaches the z_E axis) and at $\alpha_r \geq 30°$ the largest error between the no-wind state and full-wind state ($\approx 3ms^{-1}$) was 1 gram[†] [33].

The thrust variance effect also induces a variance in the control moments of the quadrotor, which is not considered in the simple model. Moreover, the simple moment model described in chapter 4 (eq. 4.13) has been found to be inadequate for predicting the pitching moment during forward flight [6]. Unfortunately, models of such aerodynamic moments are seldom discussed in literature. Thus, there appears to be a lack of analytical models for the moment variance. Instead, in chapter 7, a data-driven stepwise regression approach to deriving a moment variance model is presented. Since such data-driven approaches are specific to the identified quadrotors, they will not be treated further in this section. Enthusiastic readers are instead directed to chapter 7 (specifically, section 7.1).

5.3.2 Blade flapping

Another well documented aerodynamic effect, which is significant for the attitude control of the quadrotor, is the phenomena of blade flapping [15, 36, 38, 39]. During translational flight, the advancing blade (i.e. the blade moving towards the direction of motion) sees a higher relative velocity than the retreating blade (i.e. the blade moving away from the direction of motion) [15]. Therefore, at each revolution, the blades of the rotor oscillate or 'flap' due imbalance in lift between the blades. Since the blades both advance and retreat in a given revolution, they flap once up and once down. The net result is that the rotor plane tilts away from the direction of motion, offsetting the resultant thrust vector. This also induces a moment on the drone if the effective rotor plane is offset with respect to gravity [15, 36].

[†]The quadrotor tested was the kQuadNano which weighed about 76 grams with the battery attached [33].

Moreover, the thrust imbalance also provokes an asymmetry in the induced drag of the rotor (along the $x_B - y_B$ plane) which is otherwise cancelled out during hover (assuming windless conditions). Therefore, due to the rigidity and flapping of the blades, there is also a net horizontal drag force acting on the quadrotor which may be significant for contemporary (i.e. small) quadrotors [39]. Most quadrotors also have rotor blades which are not hinged at the hub. Therefore, the blade flapping induces a moment at the rotor hub [36]. All of these additional forces and moments have implications for the attitude control of the quadrotor.

Fortunately, a lumped sum model has been formulated from helicopter literature to (approximately) account for these effects [6, 39]. For general translational motion, a simplified relation describing the flapping angle is given by eq. 5.18 where β_{flap} and $\beta^{\perp}_{flap}$ are the steady-state flapping angles of the rotor plane to the relative velocity along and perpendicular to the direction of movement respectively [39].

$$\begin{aligned} \beta_{flap} &= -\tfrac{\mu A_{1c}}{1-\frac{1}{2}\mu^2} \\ \beta^{\perp}_{flap} &= -\tfrac{\mu A_{1s}}{1+\frac{1}{2}\mu^2} \end{aligned} \tag{5.18}$$

In eq. 5.18, μ denotes the advance ratio which is defined as the ratio of the horizontal velocity of the rotor to the linear velocity of the rotor tip (see eq. 5.19; R gives the radius of the rotor) while A_{1c} and A_{1s} are positive constants [39]. Note that eq. 5.18 is derived using the virtual hinge model, which is described and experimentally verified for a quadrotor in [15]. A virtual hinge model is necessary since the flapping angle equations used in traditional helicopter literature assume that the rotor blades are hinged at the rotor hub, which is typically not the case. Assuming that the blades are hinged culminates in a somewhat linear relationship between the flapping angle and the incoming velocity for the majority of the quadrotor's flight envelope [15]. Therefore, modelling the rotor blades as hinged results in an over-prediction of the true flapping angle of the rotor plane, as demonstrated in experiments by Hoffmann et al. [15]. Instead, to more accurately model the flapping angle, Hoffman et al. [15] propose that an effective (i.e. virtual) hinge point somewhere along the blade's length be used instead, enabling the continued use of the hinged model developed in helicopter literature. However, as highlighted by Mahony et al. [39], this virtual hinge model results in a phase shift between the sine and cosine components. Fortunately, this phase shift can be captured in the constants A_{1c} and A_{1s}.

$$\mu = \frac{\sqrt{{V_x^B}^2 + {V_y^B}^2}}{\omega_z^B R} \tag{5.19}$$

For a typical quadrotor, the advance ratio (eq. 5.19) is small. This is because the linear velocity of the rotor tips is usually higher than the translational velocity of the quadrotor (since the rotors need maintain lift during motion). Therefore, the advance ratio, μ, is assumed to be small enough such that $\frac{1}{2}\mu^2$ is negligible. Under these assumptions, eq. 5.20 can be defined indicating the sensitivity of the flapping angle of each rotor, i, to the translational velocity in the body frame wherein the first, second, and third rows denote the flapping angle due to the x-,y- and z-components of the velocity respectively [39]. Note that, unlike the equation for A_{flap} in [39], eq. 5.20 defined in this paper incorporates a directional term for A_{1s} dependent on the rotor number, i, in order to be consistent with the definition of $\{B\}$ (see fig. 4.6). In this report, the x_B and y_B run *between* the propellers while in [39] these axis run *along* the quadrotor arms. Moreover, eq. 5.20 is consistent with an equivalent formula for A_{flap} used by [6], whose authors make use of the same definition of $\{B\}$.

$$A_{flap_i} = \frac{1}{\omega_{z_i}^B R} \begin{pmatrix} A_{1c} & (-1)^{i-1} A_{1s} & 0 \\ (-1)^i A_{1s} & A_{1c} & 0 \\ 0 & 0 & 0 \end{pmatrix}, i = 1, 2, 3, 4 \tag{5.20}$$

If the stiffness of the rotor is assumed to be similar to a torsional spring, then the induced drag can be modelled as directly proportional to the flapping angle [39]. This approximation is suitable for describing the first bending mode of the rotor and the small flapping angles involved [15]. Therefore, the induced drag can be expressed using eq. 5.21 where $C_{D_{induced}}$ is the induced drag coefficient matrix with elements d_x and d_y denoting the induced drag coefficients along their respective axes. Assuming that the rotors are undamaged, then $d_x = d_y$ due to the symmetry of the rotors. Note that the induced drag is found to be proportional to the velocity of the quadrotor, rather than the squared velocity as one would expect [6, 38]. This is due to the relatively low speeds in which the quadrotor operates and due to the imbalance in lift between the advancing and retreating blades [21]. However, this is not to say that this linear velocity-drag relation also holds at higher velocities. In fact, the pioneering literature (i.e [15, 38, 39]) for quadrotor blade flapping effects only consider indoor and low speed conditions. Therefore, the relationship between the induced drag and the velocity of the quadrotor should be investigated further in both outdoor environments and in the high speed regime.

$$C_{D_{induced}} \mathbf{V^B} \approx \text{diag}\left(d_x, d_y, 0\right) \mathbf{V^B} \tag{5.21}$$

Taken together, the drag coefficient matrix describing the effects of blade flapping and induced drag stemming from rotor, i, can be described using eq. 5.22.

$$C_{D_{flap_i}} = A_{flap_i} + \text{diag}\left(d_x, d_y, 0\right) \tag{5.22}$$

Subsequently, the aerodynamic forces resulting from blade flapping and induced drag can be described through eq. 5.23. Since the flapping angle and induced drag depend on the lift (i.e. thrust) produced by the rotor, the lumped model of drag is scaled with the thrust of the respective rotor, i, in eq. 5.23 [39].

$$\mathbf{F}_{a_{flap}} = \sum_{i=1}^{4} \mathbf{T}_i C_{D_{flap_i}} \mathbf{V^B} \tag{5.23}$$

Now that the (simplified) aerodynamic forces due to blade flapping are modelled, the moments induced by the blade flapping can also be considered. As aforementioned, there are two moments which arise due to the flapping of the rotor blades. The first is a moment, $M_{flap_{lon}}$, generated by longitudinal component of the offset thrust, $T_{flap_{lon}}$, due to the tilting of the rotor plane. The second is the moment generated around the motor hub due to the stiffness, κ_{flap}, of the rotor blades. These moments are given by eq. 5.24 and eq. 5.25 respectively [15, 36]. Both of these moments are a function of the blade flapping angle, β_{flap}. In eq. 5.24, l_{cg} denotes the offset between the rotor plane and the quadrotor center of gravity (c.g.).

$$M_{flap_{lon}}^B = T_{flap_{lon}}^B \sin(\beta_{flap}) l_{cg} \tag{5.24}$$

$$M_{flap_{hub}}^B = \kappa_{flap} \beta_{flap} \tag{5.25}$$

It should be noted that the lateral tilt of the thrust vector also induces moments perpendicular to the incident wind but these are cancelled out due to the counter-rotating propellers and symmetry of the quadrotor [15].

Of course, the extensions on the simple quadrotor model described in this chapter were derived assuming a nominal quadrotor operating in the low speed regime. Therefore, the aforementioned relations, and even the simple model itself, may be invalidated by damages to the quadrotor or at higher velocities. As with many aerospace vehicles, safety is paramount. For this reason, much of the current MAV research is centered around accommodating potential faults. Some methods for doing so are treated in chapter 6. Additionally, the extensions outlined in this chapter harbour several simplifications and assumptions and subsequently fail to capture some prevalent aerodynamic phenomena and interactions, especially in the high speed regime. To bridge the gap between analytical models and realistic systems, data-driven system identification techniques are often employed and are discussed in chapter 7.

6 Actuator Faults

Given that the quadrotor is an under-actuated system, maintaining stability and control in the event of damages is a challenging task. In light of the growing use of drones in the various applications, safety of the quadrotors and their surroundings are of paramount importance. This has stimulated extensive research into fault tolerant control (FTC) investigating how to maintain control of the drone despite partial, or full, failures of one, or more, rotors [40]. Such research is aimed at maintaining sufficient control such that the drone may continue its mission or land safely. In this chapter, approaches presented in literature considering partial and full actuator faults are discussed.

Much of the literature centered around partial rotor faults tend to model faults using a multiplicative fault model [18, 19, 41] or as model uncertainties/errors [4, 20, 40, 42, 43]. The principle behind these techniques is to adjust the commanded inputs (e.g. rotor speeds, thrust, rotor power etc.) such that equilibrium is maintained. This approach can be extended to partial faults across all rotors simultaneously. Although, complications arise when considering actuator saturation limits. Moreover, these techniques rely on the assumption that the quadrotor is still fully controllable, which is typically the case but is not always true. Section 6.1 discusses the approaches used in literature to accommodate partial rotor faults.

When one, or more, rotors have failed then the nominal control strategy is no longer appropriate to provide stability and control. Therefore, the control strategy itself must be modified. Literature which consider the complete loss of one, or more, rotors typically surrender yaw control in order to maintain positional control resulting in the so-called 'relaxed' hover condition [1, 2, 21, 22]. Given the difficulty of this task, control with only single rotor failures and diagonal double rotor failures have been demonstrated through flight tests. The techniques which facilitate this control are discussed in section 6.2.1 and section 6.2.2 for the single and double rotor failure scenarios respectively. In theory, the work of Mueller et al. [22] suggests that it is also possible to maintain control with adjacent double rotor failures and even three rotor failures. However, this has not been demonstrated in flight tests and is therefore not discussed extensively here.

6.1 Partial loss of rotor effectiveness

There are two commonly taken approaches to account for partial rotor failures in literature. The first is to model the partial failure and use a fault detection algorithm to identify the occurrence of a fault (see, for example, [18, 41, 19, 44, 45]) as discussed in section 6.1.1. For such cases, partial rotor failures are typically modelled as a product of the nominal (i.e. commanded) rotor speed and a factor denoting the severity of the failure. An alternative approach is to circumvent the modelling of the failure and instead treat rotor failures as model uncertainties (as in [4, 20, 42, 46, 47]) or model errors (as in [40, 43, 48]) which are subsequently handled by a robust controller. Such approaches are summarized in section 6.1.2.

6.1.1 Modelling a partial actuator fault

When a rotor experiences a partial failure, its rotational speed is affected proportionally to the magnitude of the fault. For the i^{th} rotor ($i = 1, ..., 4$), this relationship can be expressed through eq. 6.1 where ${\omega_i^*}^B$ denotes the true rotational rate of the rotor while ${\omega_i}^B$ denotes the commanded rotational rate. $\eta_i \in (\hat{\eta}, 1]$ describes the multiplicative fault with $\eta_i = 1$ denoting a healthy rotor, $\eta_i > 0$ representing a rotor experiencing loss of effectiveness and $\eta_i = 0$ indicating a failed rotor. Here, a lower bound $\hat{\eta} > 0$ is enforced to ensure full controllability of the quadrotor [18]. Complete rotor failures are treated later on in this chapter, since these require a modification of the control strategy to maintain flight.

$$\omega_i^{*B} = \eta_i {\omega_i}^B \tag{6.1}$$

While $\hat{\eta} > 0$ provides a theoretical lower bound, Avram et al. [19] propose a more practical lower bound for $\hat{\eta}$ based on the saturation limits of the rotor. This condition is presented in eq. 6.2. For generality, it is assumed here that the rotors may have different saturation limits, denoted by $\omega_{sat,i}$ (e.g. aft rotors in fig. 4.6 may have higher saturation limits to facilitate sustained high speed flight in the wake of the front rotors [6]). However, for the conventional quadrotor, these saturation limits are the same for each rotor (i.e. $\hat{\eta}_i = \hat{\eta}\,\forall\, i \in \{1, 2, 3, 4\}$).

$$\hat{\eta}_i > \frac{1}{2\omega_{sat,i}} \sqrt{\frac{m||\mathbf{g}||}{\kappa_0}} \tag{6.2}$$

These rotor speeds directly affect the control forces and moments defined in section 4.2 (recall eq. 4.12 and eq. 4.13 respectively). Assuming that the quadrotor is operating at near-hover conditions when the fault occurs, i.e., that the thrust model given by eq. 4.11 holds, then the thrust produced by the i^{th} can be written as [18, 44]

$${T_i}^B = \left(1 - H(t - t_{f_i})\upsilon_{f_i}\right) \kappa_0 {\omega_i^2}^B \tag{6.3}$$

In eq. 6.3, $H(t - t_{f_i})$ indicates the occurrence (and evolution, in the case of time-varying faults) of a fault for rotor i at time t_{f_i}. Avram et al. model $H(t - t_{f_i})$ as a step function in both single [18] and multiple [19] fault scenarios. As with eq. 4.11, κ_0 denotes the rotor thrust coefficient. The entity $\upsilon_{f_i} \triangleq 1 - \eta_i^2$ denotes the fault parameter associated with ${\omega_i}^B$ [18]. It follows that $\upsilon_{f_i} = 0$ for a nominal rotor.

Similarly, using the control moment structure of the simple quadrotor model (eq. 4.13 in section 4.2), the complete control moment may be expressed through eq. 6.4 [18]. Here, τ_0 denotes the torque constant, b and ℓ are part of the quadrotor geometry and are defined in fig. 4.6. The 4×4 identity matrix is given by $I_{4\times4}$ and Λ_f establishes the location of an actuator fault and is therefore also a 4×4 matrix with all zero entries aside from a 1 on the diagonal corresponding to the faulty rotor [18]. For example, if rotor three experiences a fault, then $\Lambda_f = \mathrm{diag}(0, 0, 1, 0)$.

$$\mathbf{M}_c^B = \begin{bmatrix} b\kappa_0 & -b\kappa_0 & -b\kappa_0 & b\kappa_0 \\ \ell\kappa_0 & \ell\kappa_0 & -\ell\kappa_0 & -\ell\kappa_0 \\ -\tau_0 & \tau_0 & -\tau_0 & \tau_0 \end{bmatrix} \left(I_{4\times4} - H(t - t_{f_i})\upsilon_{f_i}\Lambda_f\right) \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{6.4}$$

Note that in fault model derivation of [18, 19], the rotor inertia and gyroscopic moment effects described earlier (see section 5.1 in chapter 5) are not taken into account. In [18], these effects are attenuated by the model uncertainty bounds of the fault detection and diagnosis module while in [19] a robust controller accounts for these modelling errors. However, since a model of the gyroscopic moments has already been derived in this paper, the influence of the loss of effectiveness of a single rotor can also be determined.

Recall from eq. 5.3 that the gyroscopic moment depends on both the angular rate, ω, and it's derivative, $\dot{\omega}$. Thus, taking the derivative of eq. 6.1 yields

$$\dot{\omega}_i^{*B} = \dot{\eta}_i {\omega_i}^B + \eta_i \dot{\omega}_i^B \tag{6.5}$$

Therefore, the gyroscopic moment induced by the i^{th} rotor can be represented by eq. 6.6 with $\dot{\nu}_{f_i} \triangleq 1 - \left(\dot{\eta}_i \frac{\omega}{\dot{\omega}} + \eta_i\right)$. Attention should be directed towards $\dot{\nu}_{f_i}$, which is undefined when the rotor is not accelerating (i.e. $\dot{\omega}^B = 0$). However, the last row in eq. 6.6 (containing $\dot{\nu}_{f_i}$) is also zero in such cases. Further note the difference between v_{f_i} in eqs. 6.3 and 6.4 (functions of η_i^2) and $\nu_{f_i} \triangleq 1 - \eta_i$ here (function of η_i). For cases where the loss of effectiveness factor, η_i, is not time-varying after a fault occurs, then the term $\dot{\nu}_{f_i}$ simplifies to ν_{f_i}.

$$\mathbf{M}_{r_i}^B = \begin{bmatrix} I_{r_{zz}}\Omega_y(1 - H(t - t_{f_i})\nu_{f_i})\omega_{i_z}^B \\ -I_{r_{zz}}\Omega_x(1 - H(t - t_{f_i})\nu_{f_i})\omega_{i_z}^B \\ I_{r_{zz}}(1 - H(t - t_{f_i})\dot{\nu}_{f_i})\dot{\omega}_{i_z}^B \end{bmatrix} \tag{6.6}$$

The challenge with the partial actuator fault model described by eqs. 6.3 and 6.4 is the approximation of the loss of effectiveness (i.e. identification of η_i). Typically, this is done through a fault detection and estimation algorithm, as done in [18, 41, 44], or is integrated with the adaptive control law directly [19]. In both cases, the controller is then able to compensate for the loss of effectiveness of the faulty rotors. For example, Avram et al. [19] maintained tracking performance in flight tests despite the loss in effectiveness in one, two, three and all rotors. This was accomplished by integrating the estimation of the fault degree in the adaptive control law [19]. In fact, some literature sources even avoid the fault modelling altogether and task a robust controller with accounting for partial faults.

6.1.2 Using robust control to account for failures

A popular approach to account for actuator damages is to employ a fault-tolerant, or otherwise robust, controller. The core idea being that any damages incurred by the quadrotor are treated as model uncertainties and/or errors and can subsequently be compensated for by the robust controller [1]. Therefore, the challenging task of modelling the effects of actuator damage is avoided completely. This raises the question on whether it is even worthwhile identifying a model of the damaged quadrotor, if such a model appears to be unnecessary. However, these controller-based approaches also harbour their own limitations - differing depending on the control method employed - some of which will be discussed in this section. In addition, more accurate models lessen the burden on, and facilitate the design of more adept, controllers for the given task and are therefore valuable. While the linear robust control field sees substantial interest and research, it will not be discussed extensively here due to the limitations of such methods when applied to highly non-linear systems, such as the quadrotor.

Among the robust non-linear controllers, the Sliding Mode Controller (SMC) or Sliding Mode Disturbance Observer (SMDO) sees frequent use in literature. This is primarily due to its robustness against uncertainties and (matched) disturbances [4, 20, 49, 50]. A matched disturbance is an external disturbance which enters the system through the control input channel while an unmatched disturbance is one which enters the system through other channels [51]. Since (partial) actuator faults directly influence the control input, they can be considered a matched disturbance. In more realistic scenarios where unmatched disturbances also act on the system, the SMC may be augmented with other controllers catered to handle such disturbances, as is done with the Attractive Ellipsoid Method (AEM) in [52].

Sliding Mode Control

Due to the prevalence of the SMC in literature, it will be discussed in more detail here. Given a non-linear system, the goal of the sliding mode control (SMC) is to drive the state trajectory to, and maintain the state trajectory at, a pre-defined manifold. Therefore, the design of an SMC involves [20]:

SMC-01 The selection of a manifold, or sliding (hyper)surface, for which the system exhibits the desired behaviour when confined to this manifold

SMC-02 The design of the control law such that the system approaches and remains in the manifold defined in step *SMC-01* thereafter.

In conventional SMC design, a sliding variable, σ, is defined such that when $\sigma = 0$ is satisfied, the system is 'sliding' along the desired manifold and consequently the target states are reached asymptotically [20, 50]. The design on this sliding variable is such that it can be driven to zero by the control input, u, in finite time. Conventionally, σ is defined as a function of the state error, e, between the desired states and the true states such that the desired tracking performance is assured (i.e. $e \to 0$ when $\sigma = 0$). To illustrate this, consider a system defined by eq. 6.7 where $\mathbf{X}$ is the state and $\mathbf{u}$ the control input.

$$\ddot{\mathbf{X}} = f(\mathbf{X}) + h(\mathbf{X})\mathbf{u} \tag{6.7}$$

The error, e, between desired state, $\mathbf{X}_{des}$, and true state, $\mathbf{X}$, is given by eq. 6.8.

$$\mathbf{e} = \mathbf{X}_{des} - \mathbf{X} \tag{6.8}$$

There are various ways to define the sliding variable, σ, with a common choice denoted by eq. 6.9 [20, 50]. This leads to the convergence of σ to zero with time and rate of convergence dictated by parameter c [20].

$$\sigma = \dot{\mathbf{e}} + c\mathbf{e}, \quad c > 0 \tag{6.9}$$

Note that other choices of defining σ exist, and depend on the desired performance of the error dynamics. For example, Li et al. [42] use eq. 6.10, which augments eq. 6.9 with an integral term for improved steady-state performance. Whereas others, such as [4], ignore the derivative term in eq. 6.10.

$$\sigma = \dot{\mathbf{e}} + c\mathbf{e} + k\int e \tag{6.10}$$

Regardless of the definition of the sliding variable, the subsequent process in defining the control law is the same. To remain on the sliding manifold (i.e. to maintain $\sigma = 0$), the so-called equivalent controller is used which can be derived by considering the Lyapunov Function in eq. 6.11 [50].

$$V = \frac{1}{2}\sigma^2 \tag{6.11}$$

It follows that, the sliding phase is achieved when $V = 0$. However, to guarantee asymptotic stability of the sliding phase (i.e. to remain on the sliding manifold) the following conditions need to be satisfied [50]:

i. $\dot{V} < 0$ when $\sigma \neq 0$

ii. $\lim_{|\sigma| \to \infty} V = \infty$

The derivative of eq. 6.11 is given by

$$\dot{V} = \sigma\dot{\sigma} \tag{6.12}$$

Therefore, the equivalent control, $\mathbf{u}_{eq}$, is obtained by setting $\dot{V} = 0$ and substituting $\dot{s} = \ddot{e} + c\dot{e} = \ddot{X}_d - \ddot{X} + c\dot{e}$ (with $\ddot{X}$ given by eq. 6.7) into eq. 6.12 and solving for $\mathbf{u}$.

The above derivation establishes how to stay on the sliding manifold, but not how to reach it from a point outside this manifold. To achieve this, another control law, $\mathbf{u}_{dis}$, is defined to force the trajectory towards the manifold. This can be accomplished through a simple sign function such as eq. 6.13 [50]

$$\mathbf{u}_{dis} = \text{sign}(\sigma) = \begin{cases} 1, & \text{if} \quad \sigma > 0 \\ 0, & \text{if} \quad \sigma = 0 \\ -1, & \text{if} \quad \sigma < 0 \end{cases} \tag{6.13}$$

Thus the sliding mode control law is the summation of the equivalent controller, $\mathbf{u}_{eq}$, with the discontinuous controller, $\mathbf{u}_{dis}$.

However, there is no guarantee that the system can reach the sliding manifold from the current state within finite time. In fact, the trajectory of the system can only reach the sliding surface within finite time if the reachability condition is satisfied. For the current example, this is given by eq. 6.14 where α is a parameter which dictates the reaching time (larger α corresponds to faster reaching times) [50].

$$\sigma\dot{\sigma} \leq -\frac{\alpha}{\sqrt{2}}\,|\sigma| \tag{6.14}$$

In light of this, it is important to emphasize that the SMC is only resilient to disturbances during the sliding phase and not the reaching phase [50]. Some methods, such as the Integral Sliding Mode, have been developed to mitigate matched disturbances acting on the reachable phase [50, 53]. Another issue with SMC is that, even if the reachability condition is satisfied, it does not guarantee that the system can reach the manifold safely, i.e., that the trajectory from the current state to the sliding surface is even possible.

Assuming that the sliding mode is reachable by the system, there are still some characteristics of SMC which impede its practical use. An especially notable one for quadrotor actuator faults is that SMC gains (e.g. k in eq. 6.10) increase with the uncertainty bounds or derivatives thereof [4], which may be quite substantial depending on the fault severity. High gains are undesirable since they magnify the noise in the system, could potentially excite unmodeled dynamics, and provoke the chattering phenomenon (i.e. oscillations about the sliding mode) [4, 49].

Recall that some of the main motivations for using SMC is its ability to reject disturbances and provide control, accuracy and finite time convergence of the sliding variables. These very properties are facilitated through discontinuous control inputs (e.g. eq. 6.13), which achieve the desired performance through high, effectively infinite, switching frequency between the control actions [50]. In reality, the switching frequency of the control inputs is constrained by sampling frequency of the system which contributes to the chattering effect [54]. Moreover, chattering can also be induced by unmodelled dynamics, such as the simplification of fast actuator constants [54]. For practical applications, such rapid control variations are unrealistic and the subsequent oscillatory system behaviour undesirable.

One may propose, instead, to use a continuous control input which resembles its discontinuous cousin. However, by using a continuous control input, the beneficial properties of the SMC no longer hold [50]. Namely, that the convergence of the sliding variable to zero in finite time is not guaranteed and, as a consequence, the system reaches a state only in the vicinity of the sliding manifold. Another option, known as chattering attenuation, is to instead design the SMC around the derivative of the input, such that true control input is actually continuous [50, 54]. This method results in the asymptotic convergence of the sliding variable to zero leading to what is called an asymptotic sliding mode. However, this approach only reduces the chattering and does not eliminate it completely. Similar observations can be made with other methods proposed in literature, such as higher-order sliding modes, designed to attenuate the chattering [50]. Other literature sources make use of a sliding mode disturbance observer (SMDO) to quantify the model uncertainties (e.g. actuator fault) such that high SMC gains may be avoided and the chattering effect also reduced [4, 47].

Discussion of other fault tolerant control schemes

Considering these drawbacks of SMC, some literature sources prefer the use of other fault tolerant controller schemes such as model reference adaptive control (MRAC) [40, 43, 48], incremental non-linear dynamic inversion (INDI) [11, 55], and backstepping methods [24, 56, 57]. However, these approaches also harbor their own limitations.

For instance, the adaption of the control law in MRAC is governed by the error between a reference model and the true/measured states [40]. Therefore, actuator faults are accommodated in this error and subsequent modification of the controller. However, large errors induce aggressive inputs and may lead to instability. These can somewhat be mitigated through the modified MRAC, but at the cost of poorer performance (i.e. slower adaption of the controller) which itself can also be detrimental [40].

Similarly, backstepping methods also suffer from large uncertainties in both the model and the environment. Backstepping works as follows: from a known internally stable subsystem, the controller of the immediate outer loop is designed to track the desired motion while stabilizing this subsystem. By applying this method recursively, and progressing out from the inner loops, each outer subsystem is progressively stabilized [58]. The allure of backstepping is founded in its applicability to nonlinear systems and has previously been applied in passive FTC schemes for quadrotors as in [56]. However, depending on the order of the system, deriving the backstepping control law can be quite intensive [56, 59]. Further, large parametric uncertainties may result in undesirable performance and even instability, given the model dependence of this approach [57, 60]. Even if the controller were capable of accommodating such uncertainties (e.g. through integral backstepping), this usually also comes at the cost of more conservative control laws which may result in inadequate performance. This is because these methods optimize for the stability over reducing the error between the desired and true states.

The system model dependency can be reduced through an incremental (i.e. sensor-based) variant of backstepping [57, 60]. In fact, incremental backstepping has been shown, in theory [60] and simulation [57], to be able to resist uncertainties in the control effectiveness matrix to a certain degree. Other sensor-based methods, such as INDI, also rely on the controller effectiveness matrix to sustain control [1, 4, 11]. However, these incremental methods are still susceptible to poor tracking performance in the event of a failure since the actual control effectiveness matrix may diverge considerably from the modelled control effectiveness [11].

It follows then, that reducing these model uncertainties can lead to improved performance. Indeed, some literature sources employ an adaptive control law based on (potentially time-varying) fault estimation [43, 61, 62, 63]. Inherently, these methods rely on some form of system identification, e.g. through Neural Networks as in [63], to facilitate the estimation of the model/parameters. However, the performance of many of the existing fault estimation algorithms deteriorates under conditions with external disturbances [64] or time-varying faults [43]. This is a considerable concern for outdoor applications wherein such disturbances are prevalent. While Xiao et al. [43] recently developed a more robust fault estimation algorithm, its performance was only verified in simulation. Moreover, many indoor flight tests which investigate fault tolerant control, such as [1, 3, 5, 7], make use of an external motion tracking system to provide some state information for control. Such an external motion tracking system is unrealistic for outdoor applications. Clearly, there is a need for a system identification algorithm, based only on information from the on-board sensors, which can ideally be applied online.

6.2 Complete rotor failures

As noted in section 6.1, in the event of rotor failure(s), the underactuated quadrotor is no longer fully controllable. Therefore, the control strategy itself needs to be modified in order to maintain control of the quadrotor [21]. The strategy itself may differ depending on the number and location of these failures. In literature, strategies to accommodate both single and double rotor failures have been considered, and even demonstrated through flight tests [1, 2, 21, 22, 65, 66, 67]. The general strategy is to forfeit some angular control (most commonly, yaw control) to maintain control of the altitude and reduced attitude of the quadrotor. Section 6.2.1 presents strategies for the single rotor failure case while section 6.2.2 addresses the double rotor failure case. Recent exciting research has extended current strategies from near-hover conditions to the high-speed regime [1, 2], which is also discussed in the relevant sections. Despite this, there are still some barriers to the implementation of such strategies in outdoor quadrotor applications.

6.2.1 Single rotor failure

In the event of a single rotor failure, the same equations presented in section 4.2 from chapter 4 may be used by setting the damaged rotor to zero. For instance, if rotor 4 were to have failed, then $\omega_4 = 0$. The control force, $\mathbf{F}_c^B$, and moment, $\mathbf{M}_c^B$ therefore reduce to eq. 6.15 and eq. 6.16 respectively. Note that rotor 4 is used as an example here and that the system of equations is similar given any of the other rotor failures.

$$\mathbf{F}_c^B = \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\kappa_0 \left(\omega_1^2 + \omega_2^2 + \omega_3^2\right) \end{bmatrix} \tag{6.15}$$

$$\mathbf{M}_c^B = \begin{bmatrix} b\kappa_0 & -b\kappa_0 & -b\kappa_0 \\ \ell\kappa_0 & \ell\kappa_0 & -\ell\kappa_0 \\ -\tau_0 & \tau_0 & -\tau_0 \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \end{bmatrix} \tag{6.16}$$

Definition 6.2.1

The relaxed hover conditions are those wherein the quadrotor remains substantially in the same position where the angular velocity may be non-zero. However, the *average* of the translational acceleration is zero to maintain, somewhat, the same position in space in the absence of wind [22].

It is clear that the under-actuated quadrotor is no longer able to maintain full control. Indeed, when one rotor fails, the moment symmetry in the yaw direction is broken. As such, authors in literature propose that the yaw control be surrendered to maintain positional control [21, 22, 65] or that the opposing rotor (i.e. rotor 2 if rotor 4 fails) be switched off [66, 67]. In fact, translational tracking of the quadrotor with a failed rotor has been verified through simulations in [55, 67]. Through flight tests, the authors of [21, 22] show that a quadrotor is able to maintain its position following a rotor failure, defined as the 'relaxed hover condition' (see definition 6.2.1). More recently, authors of [1, 3] have demonstrated the high speed flight of the quadrotor despite a rotor failure through wind tunnel test flights.

In these literature examples, the control strategy is the same and is based around primary-axis attitude control [21]. In the relaxed hovering condition, the primary-axis is defined as the axis about which the compromised quadrotor freely spins and is illustrated in fig. 6.1. This axis is fixed with respect to the quadrotor body and points towards the average thrust direction [1, 21]. Let $\mathbf{n}^B = (n_x, n_y, n_z)$ denote a unit vector along the primary-axis, which can be chosen arbitrarily. In order to maintain altitude, $\mathbf{n}^B$ should oppose the direction of gravity such that the inequality given by eq. 6.17 is satisfied [1].

$$-n_z \geq m\frac{||\mathbf{g}||}{T_{max}} \tag{6.17}$$

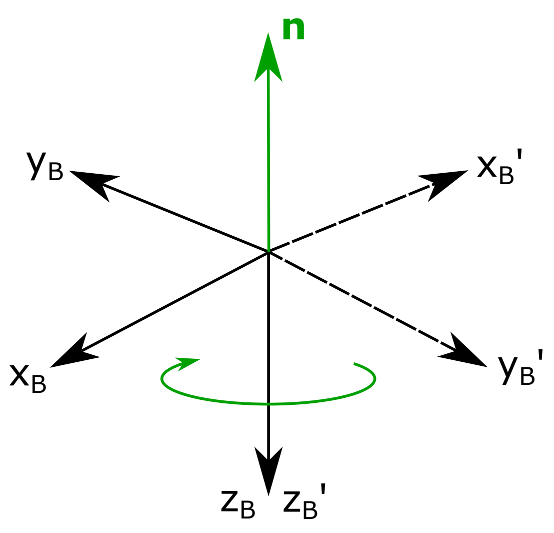


Figure 6.1: Illustration of the primary axis, $\mathbf{n}^B$, about which the quadrotor spins. For reference, the quadrotor body reference system, $\{B\}$, is also depicted for the case where $n_z = -1$ (i.e. $\mathbf{n} = [0, 0, -1]^T$). Here, the quadrotor does not wobble.

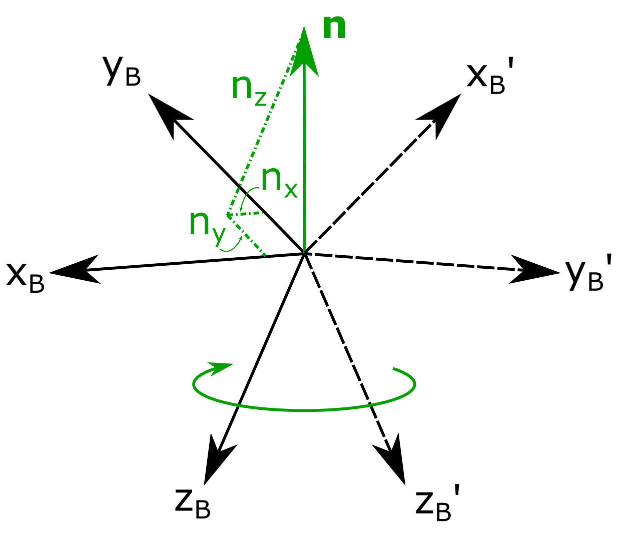


Figure 6.2: Illustration of the primary axis, $\mathbf{n}^B$, about which the quadrotor spins. For reference, the quadrotor body reference system, $\{B\}$, is also depicted for the case where $n_x > 0$ and $n_y > 0$. Here, the quadrotor wobbles due to the components of thrust in the x_B and y_B directions.

In eq. 6.17, T_{max} denotes the total available thrust. Choosing $n_z = -1$ aligns $\mathbf{n}^B$ with the instant thrust direction and results in the quadrotor spinning without wobbling (see fig. 6.1). Conversely, electing $|n_z| < 1$ means that a fraction of the generated thrust is in a direction perpendicular to gravity, thereby inducing translational motion. However, due to the yaw rotation of the quadrotor, the drone in fact follows a circular trajectory given by eq. 6.18 for a constant $\mathbf{n}^B$ [21]. Should both $n_x > 0$ and $n_y > 0$ then the quadrotor will 'wobble' while it spins [1].

$$R_p = \frac{\sqrt{1 - n_z^2}}{n_z} \frac{||\mathbf{g}||}{||\Omega^B||^2} \tag{6.18}$$

As a consequence, the quadrotor can either spin without wobbling or with wobbling (see fig. 6.2) depending on the choice of $\mathbf{n}^B$. Flight results of [21] suggest that allowing wobbling (i.e. setting $n_x > 0$ and $n_y > 0$) is more energy efficient per propeller than enforcing $n_z = -1$. However, this choice may be unacceptable given certain applications of the quadrotor (e.g. carrying a payload), but should be taken into consideration when the safety of the quadrotor is paramount.

By manipulating $\mathbf{n}^B$, the reduced attitude and position of the quadrotor can be controlled provided that $n_z \neq 0$ and the quadrotor geometries satisfy: $b \neq 0$ and $\ell \neq 0$ (see fig. 4.6). For a proof of this controllability for the single rotor condition, readers are referred to [21].

The task of the controller, then, is to define a desired acceleration in the inertial reference frame, $\mathbf{a}_{des}$, which $\mathbf{n}^B$ must track while satisfying eq. 6.17. Hence, the controller can be decomposed into a slow outer loop positional controller, which generates $\mathbf{a}_{des}$, and a fast inner loop controller whose aim is to manipulate $\mathbf{n}^B$ to achieve $\mathbf{a}_{des}$. This is accomplished by aligning $\mathbf{n}^B$ with another unit vector, $\mathbf{n}^B_{des}$, that is related to $\mathbf{a}_{des}$ through eq. 6.19.

$$\mathbf{a}_{des} = R_{EB}\left(-\frac{n_z^B}{m} T_{des}\right)\mathbf{n}^B_{des} + \mathbf{g} \tag{6.19}$$

The first term on the right hand side of eq. 6.19 denotes the projection of the desired thrust,

T_{des}, onto the current primary axis, $\mathbf{n}^B$, further projected onto the desired primary axis, $\mathbf{n}^B_{des}$. Since $\mathbf{g}$ and $\mathbf{a}_{des}$ are expressed in the inertial reference frame, R_{EB} denotes the rotation matrix from the body to the inertial reference frame. The thrust along z_E should be equal in magnitude to the gravity vector, $\mathbf{g}$, in order to maintain the altitude. Any accelerations perpendicular to gravity constitute the desired 'horizontal' movement (if any). The objective of the attitude controller is to then align $\mathbf{n}^B$ with $\mathbf{n}^B_{des}$, to be able to track the desired accelerations. In the case where $\mathbf{a}_{des} = 0$, then the quadrotor is in the relaxed hover condition with $\mathbf{n}^B$ opposing gravity.

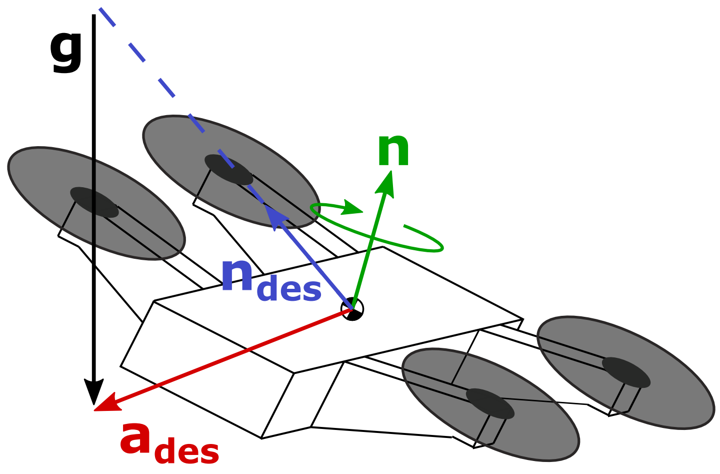


Figure 6.3: Illustration of the reduced attitude control strategy. In order to maintain altitude and move in the desired direction given by $\mathbf{a}_{des}$, the current primary axis of rotation, $\mathbf{n}$, should be aligned with the desired axis of rotation, $\mathbf{n}_{des}$.

$\mathbf{a}_{des}$ itself may be derived from a reference (i.e. commanded) position. A simple PID controller may be employed for this purpose. For instance, Sun et al. [1] implement eq. 6.20 where $(k_P, k_D, k_I > 0)$ are the PID gains for their outer loop positional control. Recall $\boldsymbol{\xi}$ and $\mathbf{V}$ are respectively the position and velocity of the quadrotor in the inertial frame, $\{E\}$.

$$\mathbf{a}_{des} = k_P(\boldsymbol{\xi}_{des} - \boldsymbol{\xi}) + k_D(\dot{\boldsymbol{\xi}}_{des} - \mathbf{V}) + k_I \int (\boldsymbol{\xi}_{des} - \boldsymbol{\xi})dt \tag{6.20}$$

All that is left is to design the inner loop reduced attitude control in order to align $\mathbf{n}^B$ with $\mathbf{n}^B_{des}$. The derivative of $\mathbf{n}_B$ is given by eq. 6.21 where $\mathbf{n}^E_{des}$ denotes $\mathbf{n}_{des}$ in the inertial reference frame, $\{E\}$. Note that both $\mathbf{a}_{des}$ and $\mathbf{g}$ are expressed in $\{E\}$. Therefore, $\mathbf{n}_{des}$ is easily obtainable in $\{E\}$ (refer to fig. 6.3).

$$\dot{\mathbf{n}}^B_{des} = -\boldsymbol{\Omega}^B \times \mathbf{n}^B_{des} + R_{BE}\dot{\mathbf{n}}^E_{des} \tag{6.21}$$

Let $\mathbf{n}^B_{des} = [d_1, d_2, d_3]^T$, then eq. 6.21 can be rewritten as

$$\begin{aligned} \dot{\mathbf{n}}^B_{des} &= \mathbf{n}^B_{des} \times \boldsymbol{\Omega}^B + R_{BE}\dot{\mathbf{n}}^E_{des} \\ &= \begin{bmatrix} 0 & -d_3 & d_2 \\ d_3 & 0 & -d_1 \\ -d_2 & d_1 & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} + R_{BE}\dot{\mathbf{n}}^E_{des} \end{aligned} \tag{6.22}$$

From here, the control strategy depends on the controller used. A general approach is provided by Mueller and D'Andrea [22] and facilitates a linear time-invariant controller design. However, rather than working with $\mathbf{n}^B_{des}$ directly, Mueller and D'Andrea [22] instead introduce a new 'control' coordinate frame, $\{C\}$, which satisfies

$$\mathbf{n}^C_{des} = R_{CB}\mathbf{n}^B_{des} = [0, 0, 1]^T \tag{6.23}$$

Whereby the rotation matrix, R_{CB}, denotes the rotation matrix from $\{B\}$ to $\{C\}$. Note that the elements of R_{CB} which correspond to a rotation about n_B may be chosen arbitrarily. The projection of the angular velocities in $\{C\}$ may likewise be obtained through eq. 6.24.

$$\mathbf{\Omega}^C = R_{CB}\mathbf{\Omega}^B = [p^C, q^C, r^C] \tag{6.24}$$

Similar to eq. 6.21, $\dot{\mathbf{n}}^C_{des}$ may be expressed as

$$\dot{\mathbf{n}}^C_{des} = -\mathbf{\Omega}^C \times \mathbf{n}^C_{des} + R_{CE}\dot{\mathbf{n}}^E_{des} \tag{6.25}$$

Let $\mathbf{n}^C_{des} = [h_1, h_2, h_3]^T$. The objective, then, is to drive $h_1 \to 0$, $h_2 \to 0$, $h_3 \to 1$, and $\dot{\mathbf{n}}^C_{des} \to 0$. Note that control of h_1 and h_2 guarantees control of h_3 since $\mathbf{n}^C_{des}$ is a unit vector, hence only h_1 and h_2 need to be controlled. Therefore, the deviation of the state vector from these targets can be summarized in eq. 6.26.

$$\boldsymbol{\epsilon} = [h_1, h_2, p^C, q^C, (r^C - \mathbf{n}^C\mathbf{\Omega}^C)]^T \tag{6.26}$$

In order to maintain hover, the altitude of the quadrotor needs to be an input variable. This leaves $N_p - 1$ variables to control the reduced attitude, where N_p is the number of functional propellers. So long as there is one functional propeller, Mueller and D'Andrea [22] propose a general approach to maintain control of the quadrotor by transitioning into the relaxed hovering condition. This approach linearizes the system about the hover solution, which vary depending on the number of failed rotors. In [22], Mueller and D'Andrea consider single, double and triple rotor failures for a conventional quadrotor. In addition to this linearization, [22] assumes that the average desired vehicle acceleration (i.e. eq. 6.19) is constant. Thus, $\dot{\mathbf{n}}^E_{des} = 0$. In fact, if the dynamics of the outer loop (i.e. positional controller) are sufficiently slow, then $\dot{\mathbf{n}}^E_{des}$ may be neglected entirely [21, 22]. Taking these considerations into account yields the following system

$$\dot{\boldsymbol{\epsilon}} \approx \begin{bmatrix} 0 & \pm||\mathbf{\Omega}^C|| & 0 & -1 & 0 \\ \mp||\mathbf{\Omega}^C|| & 0 & 1 & 0 & 0 \\ 0 & 0 & a_{11} & a_{12} & a_{13} \\ 0 & 0 & a_{21} & a_{22} & a_{23} \\ 0 & 0 & a_{31} & a_{32} & a_{33} \end{bmatrix} \boldsymbol{\epsilon} + \mathbf{Bu} \tag{6.27}$$

In eq. 6.27, a_{ij} can be obtained through the projection of eq. 4.6 on $\{C\}$. The exact values of $\mathbf{B}$ depend on the choice of $\mathbf{u}$ (i.e. choice of inputs; be this thrust, rotor speeds, power etc.) but ultimately also follow from eq. 4.6 [22].

Linearizing about the hover solution restricts the quadrotor to near-hover conditions. Instead, both [1] and [55] make use of a Non-linear Dynamic Inversion (NDI) controller to avoid this linearization. In fact, one of the objectives of [1] was to investigate the aerodynamic effects which occur during high-speed flight of a quadrotor with a failed rotor. The challenge with using NDI, however, is that the matrix in eq. 6.22 is non-invertible. Recall that $\mathbf{n}^B_{des}$ is a unit vector, therefore $\sqrt{d_1^2 + d_2^2 + d_3^2} = 1$. Consequently, accurate tracking of any two variables from d_1, d_2 or d_3 will guarantee tracking of the final variable [1]. If d_1 and d_2 are taken, then the sign of d_3 can also be inferred assuming that $\mathbf{n}^B$ is always in a direction opposing gravity. Therefore, taking the subspace given by eq. 6.28 facilitates the standard NDI procedure since the matrix is now invertible [1, 55]. In eq. 6.28, $\hat{\dot{\mathbf{n}}}^E_{des}$ represents the x and y components of $R_{BE}\dot{\mathbf{n}}^E_{des}$.

$$\begin{bmatrix} \dot{d}_1 \\ \dot{d}_2 \end{bmatrix} = \begin{bmatrix} 0 & -d_3 \\ d_3 & 0 \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} + \begin{bmatrix} d_2 \\ -d_1 \end{bmatrix} r + \hat{\dot{\mathbf{n}}}^E_{des} \tag{6.28}$$

Now a distinction is made between [1] and [55]. Lu and van Kampen [55] take the intuitive approach by defining $\mathbf{n}^B$ as the state and $\mathbf{n}^B_{des}$ as the reference. However, as uncovered in the experiments of Sun et al. [1], this choice results in poor stability performance due to measurement noise and model uncertainties. Instead, desirable stability characteristics are achieved with the opposite definition, whereby $\mathbf{n}^B$ is taken as the reference and $\mathbf{n}^B_{des}$ the state.

It should be noted that, for the quadrotor, the uncontrolled yaw rate is independent of the roll and pitch rates [21] and can therefore be obtained through the quadrotor model (i.e. third row of eq. 4.6) or through measurements. From the high-speed wind tunnel experiments of Sun et al. [1], it is preferable to use the measurements since the yaw rate varies as a function of the flight speed and heading angle, which are unaccounted for in the simple quadrotor model (i.e. eq. 4.6). Given that eq. 6.28 depends on the yaw rate, accurate estimations of this quantity are necessary for acceptable control.

Solving for the (controllable) angular rates, and replacing the left hand side of the equation with the virtual control input, $\mathbf{v}_1$, yields

$$\begin{bmatrix} p_{des} \\ q_{des} \end{bmatrix} = \begin{bmatrix} 0 & -1/d_3 \\ 1/d_3 & 0 \end{bmatrix} \left(\mathbf{v}_1 - \begin{bmatrix} d_2 \\ -d_1 \end{bmatrix} r - \hat{\dot{\mathbf{n}}}^E_{des} \right) \tag{6.29}$$

with virtual input, $\mathbf{v}_1$, given by eq. 6.30 with gains $k_x, k_y > 0$.

$$\mathbf{v}_1 = \begin{bmatrix} \dot{n}^B_x + k_x(n^B_x - d_1) \\ \dot{n}^B_y + k_y(n^B_y - d_2) \end{bmatrix} \tag{6.30}$$

In fact, eq. 6.30 simplifies to eq. 6.31 since the primary axis, $\mathbf{n}$, is fixed with respect to the body of the quadrotor. Therefore, $\dot{n}^B_x = \dot{n}^B_y = 0$.

$$\mathbf{v}_1 = \begin{bmatrix} k_x(n^B_x - d_1) \\ k_y(n^B_y - d_2) \end{bmatrix} \tag{6.31}$$

From here, both [1] and [55] implement an incremental non-linear dynamic inversion (INDI) controller to translate the desired pitch rate, roll rate, and total thrust (i.e. T_{des} in eq. 6.19) into the commanded rotor speeds, ω_i. INDI is elected since it is a sensor-based method, and thus only relies on a control effectiveness model. Therefore, the complex modelling of aerodynamic effects due to the failure of a rotor, which are unaccounted for in the simple hovering moment model of [21], can be circumvented. This also facilitates the investigation of the influence of these aerodynamic effects. Understanding these effects can better inform future quadrotor and controller designs [1].

High-speed flight tests of Sun et al. [1] show that the aerodynamic moments, $\mathbf{M}_a$, and forces, $\mathbf{F}_a$, are significant in comparison on the relaxed hovering conditions. Furthermore, the spin-induced aerodynamic moments vary in magnitude over one rotation of the quadrotor. It was found that, to account for these moments, the rotor speeds also oscillate within one rotation of the quadrotor [1]. This is in contrast to the relaxed hovering condition wherein these rotor speeds remain constant within a revolution [21, 22]. The consequence of this rotor speed variation is that the primary axis, $\mathbf{n}^E$, oscillates around the desired primary axis, $\mathbf{n}^E_{des}$, leading to tracking errors. Positional control is only sustained since the average thrust may still be aligned with $\mathbf{n}^E_{des}$ [1].

As discussed in [21, 22], variations in $\mathbf{n}^E$ also result in changes to the wobbling angle and yaw rate. Therefore, the oscillatory behaviour of the rotor speeds also induces a periodic variation in the yaw rate. Furthermore, this variation is also a function of the air speed. As expected, in low-speed conditions, the yaw rate variation is not so pronounced. However, for speeds of $4ms^{-1}$ or higher, this variation is significant.

Therefore, Sun et al. [1] found the moment model, given by eq. 6.32, to be highly nonlinear due to the aerodynamic moment acting on the quadrotor. The aerodynamic moment, $\mathbf{M}_a$, itself is a function of the airspeed, $\mathbf{V}$, heading angle, ψ, and rotational rate of the quadrotor, $\boldsymbol{\Omega}$, with

$\psi = 0$ when the quadrotor is facing the wind [1]. While not impossible, obtaining an accurate model of $\mathbf{M}_a$ is ambitious and designing a controller around this perhaps impractical [1].

$$\mathbf{M} = \mathbf{M}_c^B + \sum_{i=1}^{4} M_{r_i}^B + \mathbf{M}_a(\mathbf{V}, \psi, \mathbf{\Omega}^B) \tag{6.32}$$

With regards to the aerodynamic forces induced by the spinning, $\mathbf{F}_a$, Sun et al. [1] found a linear relationship between drag (along the incoming wind) and velocity. This is consistent with the linear relationship during the relaxed hover conditions and can be expressed through eq. 6.33 [21, 22]. In eq. 6.33, the superscript S denotes the stability reference frame, $\{S\} = \{O_S, x_S, y_S, z_S\}$, which is defined with origin coincident with O_B, x_S pointing towards the incoming wind, y_S, to the right and z_S pointing downwards. $C_{d,x}$ denotes the drag coefficient. Moreover, Sun et al. [1] found a quadratic relationship between the velocity and lateral force (i.e. along y_S) whose sign depends on the direction of spin of the quadrotor. Therefore, a model for the lateral force can be given by eq. 6.34 where r denotes the yaw rate and $C_{1,y}$ and $C_{2,y}$ are constant coefficients which can be identified from flight test data [1].

$$F_x^S = C_{d,x} V^S \tag{6.33}$$

$$F_y^S = \text{sign}(r) \left(C_{1,y} V^S + C_{2,y} {V^2}^S \right) \tag{6.34}$$

It should be noted that during the flight tests of Sun et al. [1], state information was collected using an external motion capturing system (OptiTrack) and was subsequently transmitted to the quadrotor via WiFi. This is due to limitations in the on-board sensing hardware coupled with the challenges of state estimation under the relaxed hover condition (i.e. when the quadrotor is spinning). Therefore, the first step in transitioning to outdoor applications is to demonstrate these novel techniques using only the on-board sensors. This challenge is only compounded by the fact that, outdoors, external disturbances are more prevalent in comparison to a more controlled indoor environment (such as a wind tunnel). To this end, Solanki [16] has recently developed a state estimation routine capable of attitude estimation during single rotor failures using only the on-board sensors, however this has not been verified outdoors. Indeed, some literature has begun the transition into outdoor experiments, such as that of Stephan et al. [68] on fault tolerant control using linear parameter-varying control. However, [68] considers near-hover and low speed scenarios ($\approx 1ms^{-1}$), and does not describe the external conditions of the outdoor flight tests.

Furthermore, it is commonly assumed in current literature on complete rotor failures that the entire rotor is removed once the failure occurs [1, 2, 3, 21]. Therefore, there are no effects of an idle or otherwise windmilling rotor. In practice, a failed rotor may remain attached to the quadrotor and thus subsequent aerodynamic effects and interactions will be present. Therefore, these effects should also be accounted for in future research.

6.2.2 Double rotor failure

When two rotor failures are considered in literature, the constraint that these failures occur diagonal to each other on the quadrotor is often imposed [21, 22, 67]. In fig. 4.6, this would entail rotors 1 & 3 or rotors 2 & 4 failing together. However, some sources also consider adjacent propeller failures, which is more likely than diagonal failures, and even extend their methods to three propeller failures [21, 22]. Although, such techniques are only verified in simulation and have yet to make the leap to flight tests. Conversely, control of a quadrotor with diagonal rotor failures has been demonstrated in flight tests both in hovering [21, 22] and high-speed [2] conditions.

Similar to the single rotor failure case in section 6.2.1, a quadrotor with opposing double rotor failures can also be controlled by surrendering yaw control [21, 22]. Therefore, control task is again to align the primary axis (i.e. axis of rotation), $\mathbf{n}^B$, with a desired primary axis, $\mathbf{n}^B_{des}$ derived from the positional controller (i.e. $\mathbf{n}^B_{des}$ which maintains the altitude and moves in the desired direction with acceleration $\mathbf{a}_{des}$) through the remaining inputs, $\mathbf{u}$.

The control input definition depends on which rotors have failed. If rotors 2 & 4 have failed, then the control input can be described by

$$\mathbf{u} = [u_1, u_2]^T = [\omega_1^2, \omega_3^2]^T \to s_l = 1 \tag{6.35}$$

Whereas, if rotors 1 & 3 fail instead, then the control input is

$$\mathbf{u} = [u_1, u_2]^T = [\omega_2^2, \omega_4^2]^T \to s_l = -1 \tag{6.36}$$

Since the dynamics (e.g. the yaw rate direction) depend on the type of failure, the variable $s_l \in \{-1, 1\}$ is defined to represent this. For a quadrotor with the same configuration as fig. 4.6, let $s_l = 1$ when only rotors 1 & 3 remain functional while $s_l = -1$ if only rotors 2 & 4 are operational.

Again, Mueller and D'Andrea [21, 22] propose the same generalized approach linearized about the hovering conditions mentioned when discussing single rotor failure (section 6.2.1). In essense, control is facilitated by setting the remaining rotors to constant speeds that are different between the rotors (e.g. $\omega_2 > \omega_4$, $\dot{\omega}_2 = \dot{\omega}_4 = 0$) [22]. However, to facilitate the movement of the drone in high-speed conditions, the approach of [2] is presented in depth here. Note that this approach is based on the authors' prior work on single rotor failure [1, 55] and thus the same equations may be employed. Recall eq. 6.22 with $\mathbf{n}^B_{des} = [d_1, d_2, d_3]^T$, which is reiterated here for convenience.

$$\dot{\mathbf{n}}^B_{des} = \begin{bmatrix} 0 & -d_3 & d_2 \\ d_3 & 0 & -d_1 \\ -d_2 & d_1 & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} + R_{BE}\dot{\mathbf{n}}^E_{des}$$

In contrast to the single rotor failure case, it is most energy efficient to spin without wobbling during hover [21]. Therefore, $\mathbf{n}^B$ should be aligned with the instantaneous thrust direction (i.e. $\mathbf{n}^B = [0, 0, -1]^T$). Accordingly, the goal of the controller is to bring $d_1 = d_2 = 0$ for hovering conditions. Given that $\sqrt{d_1^2 + d_2^2 + d_3^2} = 1$, and assuming $\mathbf{n}^B$ always points in a direction opposing gravity, it follows that tracking on d_1 and d_2 also guarantees tracking of d_3.

When operating in the high-speed flight regime, complex aerodynamic effects become increasingly significant. To account for this, Sun et al. [2] define the 'relaxed trimming equilibrium' which is an extension of the relaxed hovering condition of Mueller et al. [21]. The primary difference between these being that the relaxed trimming equilibrium harbors a drag force, $F_{a,xy}$, which must also be accounted for by the average thrust of the quadrotor to maintain equilibrium. Under these conditions, the magnitude of thrust needs to be increased relative to the hovering condition, while the primary axis can remain along the thrust vector (i.e. $\mathbf{n}^B = [0, 0, -1]^T$). It becomes apparent, then, that the saturation of the rotors is a pressing concern for high-speed flight suffering actuator faults. In fact, Sun et al. [2] suspect that such actuator saturation is the cause of deteriorating tracking performance with increasing flight speed following the eventual loss of control at a speed of $8.8ms^{-1}$.

Another consideration for double rotor failures in the high-speed regime is the definition of the control inputs. As with single rotor failure in section 6.2.1, the double rotor failure also makes use of a cascaded control structure: the outer loop handles the positional control while the inner loop manages the attitude control. The challenge with two failed rotors is that only

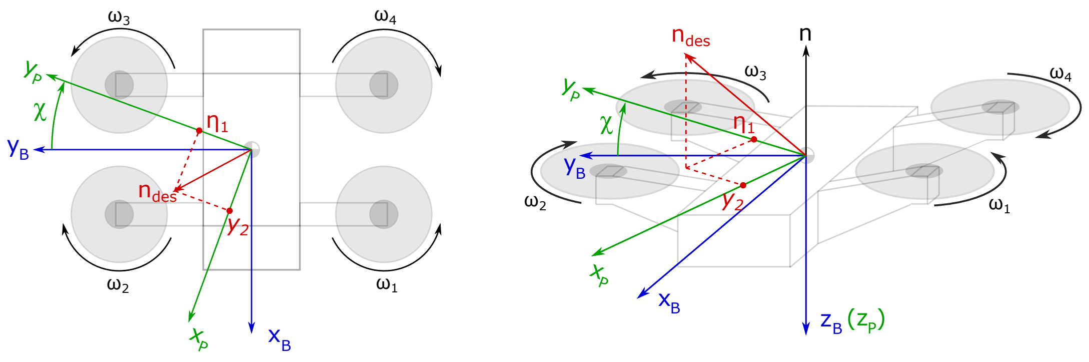


Figure 6.4: Definition of χ and the new coordinate system, $\{P\} = \{O_p, x_P, y_P, z_P\}$, on a top down view (left) and 3-Dimensional view (right) of the quadrotor.

two controlled outputs may be elected. Clearly, one of the control outputs should be the altitude in order for the quadrotor to maintain flight. Therefore:

$$y_1 \triangleq z \tag{6.37}$$

This leaves one variable to control the reduced attitude. In the case of a single rotor failure, the reduced attitude was controlled through elements d_1 and d_2 of the unit vector $\mathbf{n}^B_{des}$. With two failed rotors this is no longer possible. Instead, Sun et al. [2] propose that these quantities be controlled implicitly through a linear combination of d_1 and d_2.

By defining a new coordinate system, $\{P\} = \{O_P, x_P, y_P, z_P\}$, the second output, y_2, can be defined as a projection of $\mathbf{n}^B_{des}$ on x_P [2]. This new coordinate system is obtained by rotating the body frame, $\{B\}$, about the z_B-axis through angle χ (refer to fig. 6.4). The sign of χ depends on which two rotors are operational and can therefore be expressed through eq. 6.38.

$$\chi = s_l|\chi| \tag{6.38}$$

Therefore, the second output, y_2, can be defined as:

$$y_2 \triangleq d_1 \cos\chi + d_2 \sin\chi \tag{6.39}$$

However, this choice leaves the projection of $\mathbf{n}^B_{des}$ on y_P uncontrolled, which may induce instability. Therefore, in order to determine the conditions for which the system is stable, the inner loop state vector, $\mathbf{x}_{inner}$, can be decomposed further into external, ξ_{inner}, and internal states, η_{inner}, by transforming the nonlinear affine system into normal form [2, 69, 70]. Sun et al. define $\mathbf{x}_{inner}$ as [2]:

$$\mathbf{x}_{inner} = \begin{bmatrix} d_1 & d_2 & p & q & r & z & w \end{bmatrix}^T \tag{6.40}$$

The external states are derived from the chosen inner loop control outputs (i.e y_1 and y_2) and their first derivatives [2]. Equation 6.41 denotes the derivative of y_1.

$$\dot{y}_1 = \dot{z} = w \tag{6.41}$$

The derivative of y_2 follows from eq. 6.22 and is given by eq. 6.42 with c_1 and c_2 denoting to the first and second rows of $R_{EB}\dot{\mathbf{n}}^E_{des}$ respectively.

$$\begin{aligned} \dot{y}_2 &= \dot{d}_1 \cos\chi + \dot{d}_2 \sin\chi \\ &= (-d_3 q + d_2 r + c_1)\cos\chi + (d_3 p - d_1 r + c_2)\sin\chi \end{aligned} \tag{6.42}$$

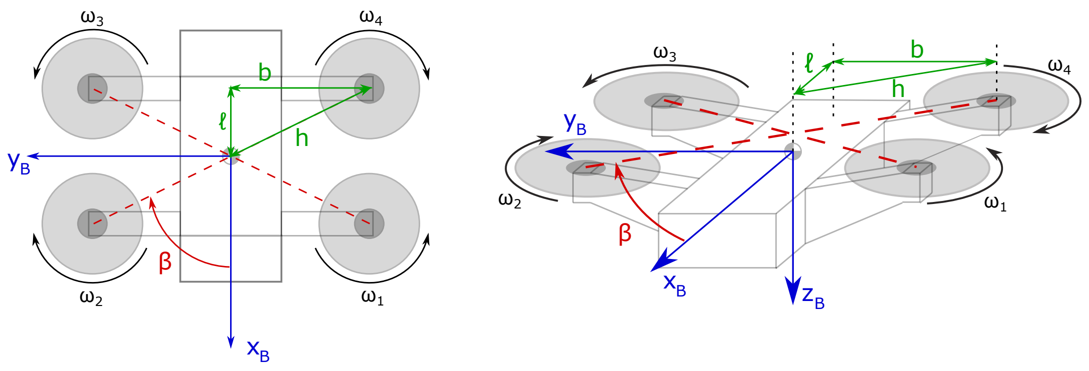


Figure 6.5: Definition of β and geometric property h on a top down view (left) and 3-Dimensional view (right) of the quadrotor.

Therefore, the external states can subsequently be described as

$$\begin{bmatrix}\xi_1\\ \xi_2\\ \xi_3\\ \xi_4\end{bmatrix} = \begin{bmatrix}y_1\\ \dot{y}_1\\ y_2\\ \dot{y}_2\end{bmatrix} = \begin{bmatrix} z\\ w\\ d_1\cos\chi + d_2\sin\chi\\ (-d_3q + d_2r + c_1)\cos\chi + (d_3p - d_1r + c_2)\sin\chi \end{bmatrix} \tag{6.43}$$

With four external states, three internal states should be defined to complete the state vector. Sun et al. [2] therefore define the projection of $\mathbf{n}^B_{des}$ on y_D as an internal state, η_1, along with two other internal states described in eq. 6.44. While the choice of internal states is up to the designer, these internal states must satisfy the conditions that their first order derivative does not contain the control input, $\mathbf{u}$, and that resulting transformation matrix (i.e. transformation of states to their normal representation) is invertible [2].

$$\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\end{bmatrix} = \begin{bmatrix}-d_1\sin\chi + d_2\cos\chi\\ q\cos\zeta - s_l p\sin\zeta\\ r + s_n\mu w\end{bmatrix} \tag{6.44}$$

The virtual angle, ζ, is defined in eq. 6.45 [2]. Similarly, eq. 6.46 describes the entity μ. The variable $s_n \in \{-1, 1\}$ depends on the direction of rotation of the remaining rotors. For clockwise spinning rotors $s_n = 1$, conversely $s_n = -1$ for counterclockwise spinning rotors. For generality, Sun et al. [2] define β as the angle between the x_B-axis and the rotor arm connecting rotors 2 & 4 with $\beta \in [0, \frac{\pi}{2}]$ (see fig. 6.5). This corresponds to $\beta = \frac{\pi}{4}$ with the definition of $\{B\}$ in fig. 4.6. The moments of inertia of the drone are given by I_x and I_y assuming that the products of inertia are negligible, i.e. $\mathbf{I_v} = \mathrm{diag}(I_x, I_y, I_z)$. The mass is given by m, τ_0 is the rotor torque coefficient and κ_0 is the thrust coefficient.

$$\zeta = \tan^{-1}\left(\frac{I_x}{I_y}\cot\beta\right) \tag{6.45}$$

$$\mu = \frac{m\tau_0}{d_3\kappa_0} \tag{6.46}$$

Sun et al. [2] demonstrate that, in the relaxed trim equilibrium, the internal dynamics of the inner-loop system are locally asymptotically stable if and only if χ is chosen such that all the eigenvalues of $\mathbf{A}$ in eq. 6.47 have strictly negative real parts.

$$\mathbf{A} = \frac{s_r}{\sin(|\chi| - \zeta)}\begin{bmatrix}-\bar{r}\cos(|\chi| - \zeta) & 1\\ -\bar{r}\Lambda & \Delta\end{bmatrix} \tag{6.47}$$

with

$$\Lambda = (A_x \bar{r} + 2a_x \bar{\omega} s_r) \sin^2 \zeta + (A_y \bar{r} - 2a_y \bar{\omega} s_r) \cos^2 \zeta \tag{6.48}$$

$$\Delta = -(A_x \bar{r} + 2a_x \bar{\omega} s_r) \sin \zeta \sin |\chi| + (A_y \bar{r} - 2a_y \bar{\omega} s_r) \cos \zeta \cos |\chi| \tag{6.49}$$

$$A_x = \tfrac{I_y - I_z}{I_x}, \quad A_y = \tfrac{I_z - I_x}{I_y}, \quad A_z = \tfrac{I_x - I_y}{I_z} \tag{6.50}$$

$$a_x = \tfrac{I_p}{I_x}, \quad a_y = \tfrac{I_p}{I_y} \tag{6.51}$$

subject to the following assumptions [2]:

assumption-01	The aerodynamic forces, $\mathbf{F}_a$, and moments, $\mathbf{M}_a$, are independent of the control input $\mathbf{u}$.
assumption-02	The desired primary axis, $\mathbf{n}^E_{des}$, varies slowly in time such that $\dot{\mathbf{n}}^E_{des} \approx 0$.
assumption-03	The average rotor speed is constant ($\omega \approx \bar{\omega}$) and independent of the input.
assumption-04	The yaw rate of the quadrotor is constant, $r \approx \bar{r}$.

For the proof of this proposition, enthusiastic readers are referred to [2].

Sun et al. [2] conclude that the stability of the internal dynamics is invariant of the failure type (i.e. $\forall\, s_l$) provided that $|\chi|$ is constant. However, this may not hold under significant aerodynamic forces and moments since these were neglected by Sun et al. [2] for this particular conclusion.

Although *assumption-03* seems rather counter-intuitive, it has been corroborated by flight test data in [2]. Recall that Sun et al. [2] consider the relaxed trim equilibrium, wherein the total thrust force is approximately constant. Therefore, *assumption-03* is only invalidated when this equilibrium is broken, such as during aggressive maneuvers. Despite this, Sun et al. [2] found the internal dynamics to be stable even when the quadrotor was tasked with simple lateral and vertical maneuvering under windless conditions. This is likely a consequence of the commanded thrust being constant for various intervals of time, rather than being continuously changing. Hence, *assumption-03* is only broken in the vicinity of a commanded thrust change (e.g. step increase), but remains valid otherwise.

As with their prior work, Sun et al. [2] again make use of an INDI controller, which is restructured to accommodate these internal dynamics. Even though the stability of the internal dynamics can be ensured with appropriate selection of χ, the performance of the controller depends on the control effectiveness matrix. The effects of the inputs, $\mathbf{u}$, on the output states, y_1 and y_2, can be seen through their second derivatives which are given by eq. 6.52 and eq. 6.53 respectively.

$$\ddot{y}_1 = ||\mathbf{g}|| + \frac{F_{a,z}}{m} - \frac{R_{33}\kappa_0(u_1 + u_2)}{m} \tag{6.52}$$

$$\ddot{y}_2 = \ddot{d}_1 \cos \chi + \ddot{d}_2 \sin \chi \tag{6.53}$$

In eq. 6.52, $F_{a,z}$ represents the aerodynamic forces acting on the quadrotor along the z_E axis, $\mathbf{g}$ is the gravity vector, and R_{33} denotes the last term on the diagonal of R_{EB}. Grouping terms, eq. 6.52 can easily be rewritten as eq. 6.54 where Φ_1 describes the forces acting on the quadrotor along the z_E-axis and B_1 denotes the control effectiveness on y_1.

$$\ddot{y}_1 = \Phi_1 + B_1(u_1 - u_2) \tag{6.54}$$

$$B_1 = -\frac{R_{33}\kappa_0}{m} \tag{6.55}$$

In contrast, the control effectiveness matrix in eq. 6.53 is not as clear. $\ddot{d}_1$ and $\ddot{d}_2$ can be obtained by taking the derivative of eq. 6.21. Using the product rule, this yields

$$\ddot{\mathbf{n}}_{des}^B = \dot{\mathbf{n}}_{des}^B \times \mathbf{\Omega}^B + \mathbf{n}_{des}^B \times \dot{\mathbf{\Omega}}^B + \mathbf{\Omega}^B \mathbf{n}_{des}^E + R_{BE}\ddot{\mathbf{n}}_{des}^E \tag{6.56}$$

Only the second term in eq. 6.56 (i.e. $\mathbf{n}_{des}^B \times \dot{\mathbf{\Omega}}^B$) contains quantities related to the input, $\mathbf{u}$, while all the other terms are only functions of the state vector, $\mathbf{x}_{inner}$.

$$\mathbf{n}_{des}^B \times \dot{\mathbf{\Omega}}^B = \begin{bmatrix} d_2\dot{r} - d_3\dot{q} \\ d_3\dot{p} - d_1\dot{r} \\ d_1\dot{q} - d_2\dot{p} \end{bmatrix} \tag{6.57}$$

The rotational accelerations, $\dot{\Omega} = [\dot{p}, \dot{q}, \dot{r}]^T$, can be obtained through the simple moment model eq. 4.6 (see section 4.2) with the addition of the aerodynamic moments acting on the quadrotor, $\mathbf{M}_a = [M_{a,x}, M_{a,y}, M_{a,z}]^T$. $\dot{\mathbf{\Omega}}$ is given by eq. 6.58.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} A_x rq - 2a_x q\bar{\omega}s_n + s_l G_p(u_1 - u_2) + M_{a,x} \\ A_y rp - 2a_y p\bar{\omega}s_n + G_q(u_1 - u_2) + M_{a,y} \\ A_z pq - \frac{\gamma r}{I_z} - s_n G_r(u_1 + u_2) + M_{a,z} \end{bmatrix} \tag{6.58}$$

In eq. 6.58, A_x, A_y, and A_z represent the inertia terms and were previously defined in eq. 6.50. Likewise, the terms a_x and a_y are given by eq. 6.51. $\bar{\omega}$ follows from *assumption-03* and denotes the average rotational rate of the rotors. The aerodynamic drag induced by the yaw rate is captured by the yaw damping coefficient, γ. This coefficient has been found to vary linearly with the yaw rate [21]. The terms G_p, G_q, and G_r relate to the thrust and torque coefficients and are presented in eq. 6.59 where h is the shortest distance from the rotors to the center of gravity (from the geometry of fig. 4.6, $h = \sqrt{b^2 + \ell^2}$. For an illustration see also fig. 6.5).

$$G_p = \frac{\kappa_0 h \sin\beta}{I_x}, \quad G_q = \frac{\kappa_0 h \cos\beta}{I_y}, \quad G_r = \frac{\tau_0}{I_z} \tag{6.59}$$

Grouping terms related to $\mathbf{u}$ and substituting the first two rows of eq. 6.56 into eq. 6.53 yields

$$\begin{aligned} \ddot{y}_2 \quad = \quad & \{\Phi_{2,1}(\mathbf{x}) + [-d_2 s_n G_r(u_1 + u_2) - d_3 G_q(u_1 - u_2)]\}\cos(\chi) \\ & +\{\Phi_{2,2}(\mathbf{x}) + [d_3 s_l G_p(u_1 - u_2) + d_1 s_n G_r(u_1 + u_2)]\}\sin(\chi) \end{aligned} \tag{6.60}$$

The lumped terms $\Phi_{2,1}(\mathbf{x})$ and $\Phi_{2,2}(\mathbf{x})$ encompass nonlinear terms and aerodynamic effects from the first and second rows of eq. 6.56 respectively. Sun et al. [2] simplify eq. 6.60 by exploiting knowledge of the controllability of the conventional quadrotor. Namely, that it requires less control effort to roll and pitch than to yaw. This also follows from the angular acceleration control effectiveness given in eq. 6.59. Thus, the following assumption can be made

$$|G_r| << \min(|G_p|, |G_q|) \tag{6.61}$$

Therefore, eq. 6.60 may be simplified further to eq. 6.62. Here, $\Phi_2(\mathbf{x}, \chi)$ represents the nonlinear terms and aerodynamic effects projected onto x_P.

$$\ddot{y}_2 = \Phi_2(\mathbf{x}, \chi) + d_3\{s_l G_p \sin(\chi) - G_q \cos(\chi)\}(u_1 - u_2) \tag{6.62}$$

Recall that $\chi = s_l|\chi|$, therefore

$$s_l \sin(s_l|\chi|) = \sin(|\chi|) \quad \forall \quad s_l \in \{-1, 1\} \tag{6.63}$$

Substituting G_p, G_q and $\chi = s_l|\chi|$ into eq. 6.62 gives

$$\begin{aligned} \ddot{y}_2 &= \Phi_2(\mathbf{x}, \chi) + d_3\{\tfrac{\kappa_0 h \sin\beta}{I_x} \sin(|\chi|) - \tfrac{\kappa_0 h \cos\beta}{I_y} \cos(|\chi|)\}(u_1 - u_2) \\ &= \Phi_2(\mathbf{x}, \chi) + \tfrac{d_3 h \kappa_0 \sin\beta}{I_x}\{\sin(|\chi|) - \tfrac{I_x \cos\beta}{I_y \sin\beta} \cos(|\chi|)\}(u_1 - u_2) \end{aligned} \tag{6.64}$$

Notice that the coefficient of $\cos(|\chi|)$ is equivalent to $\tan(\zeta)$ (refer to eq. 6.45). Therefore, eq. 6.64 can be succinctly expressed as eq. 6.65 where B_2 represents the control effectiveness matrix and is defined in eq. 6.66.

$$\ddot{y}_2 = \Phi_2(\mathbf{x}, \chi) + B_2(u_1 - u_2) \tag{6.65}$$

$$B_2 = -\frac{d_3 \kappa_0 \sin(\beta)}{I_x \cos(\zeta)} \sin(\zeta - |\chi|) \tag{6.66}$$

Therefore, to guarantee control, both B_1 and B_2 should not be zero. This results in the following conditions for controllability:

- For y_1
  i. $R_{33} \neq 0$: Thrust should be outside the horizontal plane in $\{E\}$
- For y_2
  ii. $d_3 \neq 0$: The desired primary axis should not not perpendicular to the z_B-axis (i.e. thrust must have a component along z_B)
  iii. $\sin(\zeta - |\chi|) \neq 0$: Adds an additional constraint on χ to ensure that the quadrotor is controllable in addition to being stable.

Note that as $|\sin(\zeta - |\chi|)| \to 0$, the control effectiveness on y_2 deteriorates culminating in aggressive control inputs. Such inputs are undesirable and may be unrealistic considering the actuator dynamics. Therefore, Sun et al. [2] also impose eq. 6.67 as a constraint on the selection of χ.

$$\gamma_{ce}(|\chi|) \triangleq \frac{B_2(|\chi|)}{\min(|G_p|, |G_q|)} \geq 1 \tag{6.67}$$

Through a case study of three different selections of χ:

- $|\chi_{c,1}|$: which violates the condition $\gamma_{ce}(|\chi|) \geq 1$
- $|\chi_{c,2}|$: which has positive real eigenvalues in $\mathbf{A}$
- $|\chi_{c,3}|$: which has strictly negative real eigenvalues in $\mathbf{A}$ and satisfies $\gamma_{ce}(|\chi|) \geq 1$

It was found that, indeed, choosing $|\chi_{c,1}|$ results in large and oscillatory actuator inputs which eventually results in a crash due to actuator limitations [2]. Alternatively, choosing $|\chi_{c,2}|$ exhibited a divergence of the internal states, indicating instability and eventually leading to a crash. Selecting $|\chi_{c,3}|$ results in the convergence of the states and inputs [2].

It is noteworthy to emphasize that INDI relies on measurement updates, which are subject to noise, to generate the control law. To mitigate the effects of measurement noise, Sun et al. [2] apply a low-pass filter to the variables, which induces a delay on the signal. Therefore, in practice, all time-varying variables should be filtered using the same cut-off frequency to ensure that the signals are synchronized.

While an exciting advancement in the field of quadrotor control, the contributions of Sun et al. [2] - and, in fact, many others on double rotor failures such as [21, 22] - only demonstrate[†] the control of quadrotors with diagonal rotor failures and not adjacent failures. The latter failure scenario is perhaps the most common type of double rotor failure and is therefore an important avenue in which to extend current knowledge. Indeed, Mueller et al. [22] outline a strategy for control with two adjacent failed rotors (as described in section 6.2.1). In the adjacent rotor failure case, pitch/roll control is surrendered and the remaining rotors set to a constant thrust level that is different for each rotor [22]. This thrust level is defined by the ratio of thrusts which consumes the least amount of power, and thus varies per quadrotor. However, this is geared towards finding a hovering solution. Therefore, the proposed strategy is only valid in near-hover conditions and may not be the most effective policy. Perhaps, in the event of adjacent rotor failures, movement may be facilitated by making use of the aerodynamic forces and moments acting on the quadrotor.

Furthermore, as with the single rotor failure case, the results of Sun et al. [2] were facilitated through accurate state estimates obtained from an external motion tracking system (OptiTrack). Again, these control strategies should be compatible with on-board state estimation to expedite their practical use in outdoor applications.

[†]In terms of flight tests. Mueller et al.[22] have shown that, in theory, it is possible to control a quadrotor with adjacent failures in the relaxed hover condition.

7 System identification of quadrotors

What has emerged from the previous chapters is that the development of more accurate quadrotor models is necessary to fully exploit UAVs while simultaneously improving safety. Up to now, a simple quadrotor model has been described upon which some extensions to this base model, founded on well-studied phenomena and actuator faults, were discussed. However, several simplifications were made in the derivations of these extensions. For example, in the case of actuator faults, it is often assumed that the rotor is somehow removed from the quadrotor such that complex interaction effects need not be considered. In reality, failed rotors may remain attached to the quadrotor and their influence on the controllability perhaps relevant. Quadrotors routinely operate at higher speeds where unmodelled aerodynamic effects become prevalent. Moreover, complex interaction and coupling effects emerge during aggressive manoeuvres and remain unaccounted for in the models discussed previously. Understanding how these effects influence the quadrotor, and perhaps even how to exploit them, facilitates optimized control of the (un)damaged quadrotor.

It is impractical to analytically derive models which are valid across the entire flight regime. Therefore, research into quadrotor models beyond the domain of validity of current analytical models typically make use of various data-driven system identification techniques. Exact approaches differ depending on the application and desired accuracy. This chapter discusses a few of these approaches starting with a simple, yet effective, approach using polynomial models in section 7.1. Given recent advances in Artificial Neural Networks, and their ever-growing popularity, section 7.2 outlines a Neural Network approach to system identification.

7.1 Polynomial models

One of the merits of using polynomial based models is that they are simple to apply and interpret [7]. Prior knowledge can be applied to determine both the structure and regressors of the polynomial, facilitating a so-called 'gray-box' approach to modelling [7]. Of course, polynomial models can also be employed when little information is known, although selecting appropriate model candidates becomes more challenging. Therefore, polynomial models are often used as an initial modelling technique for system identification in aerospace [71].

Polynomial models are a data-driven approach wherein measurement data are used to fit (i.e. 'train') the model. This is accomplished through regression, which is typically of the form

$$z = A\theta + \epsilon \tag{7.1}$$

where $z \in \mathbf{R}^N$ denotes the measurement data of length N. The matrix A represents arbitrary combinations (e.g. power series) of the independent variables (in this case, the quadrotor states) wherein the first column is a constant vector. This matrix is expanded in eq. 7.2. The parameters corresponding to the regressors (i.e. elements) of A are given by θ. Therefore, $A\theta$ denotes predictions of the model. The residuals between these predictions and the measurement data is denoted by ϵ.

$$A = \begin{bmatrix} 1 & \xi_{1,1} & \xi_{1,2} & \cdots & \xi_{1,m} \\ 1 & \xi_{2,1} & \xi_{2,2} & \cdots & \xi_{2,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi_{N,1} & \xi_{N,2} & \cdots & \xi_{N,m} \end{bmatrix} = [\mathbf{1}, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_m] \tag{7.2}$$

For systems which are linear in the parameters, the optimal model parameters, $\hat{\theta}$, which minimize the error residuals in eq. 7.1 can be estimated through ordinary least squares (eq. 7.3).

$$\hat{\theta} = (A^T A)^{-1} A^T z \tag{7.3}$$

Once the desired model fidelity is known, designers must decide on the appropriate model structure to satisfy the requirements. It is clear from eq. 7.1 that the model performance depends on the choice of the regressor matrix, A. The complexity of A is a function of the chosen regressors and the number of thereof. Increasing the complexity of A may lead to lower model residuals at the risk of over-fitting the measurement data. Conversely, a simple regressor matrix may fail to adequately capture the governing dynamics. Moreover, the computational load increases with the complexity of A. This therefore raises the question of how to choose a suitable model structure.

7.1.1 Stepwise regression

Sun et al. [6] propose the use of a stepwise model structure selection algorithm, which sees frequent use in conventional aircraft system identification literature [71, 72], to identify a high-speed model of a quadrotor. The motivation behind stepwise regression is to let the algorithm determine the most suitable model structure given a set of model candidates. This set is known as a candidate pool of regressors and is comprised of combinations - be these polynomials, exponents, logarithms or otherwise - of the quadrotor states.

The principle behind this approach is to, in the forward step, select a regressor from the candidate pool which leads to the greatest improvement in model accuracy. Subsequently, in the backward step, the quality of the current selected regressors is reassessed by evaluating the performance of the model with each of the regressors removed. This is done since regressors added earlier may become redundant due to the combination of the current model regressors, and may therefore be removed without detriment to the model quality. The algorithm terminates once the regressor that was removed in the backward step is the same as that added in the forward step, otherwise the forward step commences again and the process repeats itself.

The stepwise regression is initialized with eq. 7.1 and $\mathbf{A} = [1, \ldots, 1]^T$. Regressors are added to the model based on their correlation with the measurement points, $\mathbf{z}$, after adjustment for the effects of the current regressors in the model [71]. Note that, immediately following initialization, the only regressor in the model is the bias vector, $\mathbf{A} = [1, \ldots, 1]^T$. This adjustment is conducted in order to ensure that the added regressors capture variations of the measurement data which are otherwise absent in the current model (i.e. absent in $\mathbf{A}$) [71]. The adjustment is performed as follows [71]:

i. The candidate regressors in the pool are adjusted to orthogonalize them with respect to the regressors already in $\mathbf{A}$. This is accomplished through eq. 7.4.

ii. The measurement data, $\mathbf{z}$, are likewise modified using eq. 7.5 to account for the adjustments of the candidate regressors in the previous step. Note that $\mathbf{A}$ is the same in both eq. 7.4 and eq. 7.5

$$\boldsymbol{\epsilon}_{\xi,j} = \boldsymbol{\xi}_j - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{\xi}_j \tag{7.4}$$

$$\boldsymbol{\epsilon}_{z,j} = \mathbf{z}_j - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{z}_j \tag{7.5}$$

Following this adjustment, the correlation of the (adjusted) candidate regressors with the (modified) measurement points can be assessed for each regressor using eq. 7.6 [71]. The regressor corresponding to the highest absolute correlation is selected for addition to the model. Note that the original regressor is selected and not its adjusted variant. For example, if $\boldsymbol{\epsilon}_{\xi,1}$ results in the highest correlation, then regressor $\boldsymbol{\xi}_1$ is elected for model addition.

$$r_j = \frac{\sum_{i=1}^{N} \left(\epsilon_{\xi,j}[i] - \bar{\epsilon}_{\xi,j}\right)\left(\epsilon_{z,j}[i] - \bar{\epsilon}_{z,j}\right)}{\sqrt{\sum_{i=1}^{N} \left(\epsilon_{\xi,j}[i] - \bar{\epsilon}_{\xi,j}\right)^2 \sum_{i=1}^{N} \left(\epsilon_{z,j}[i] - \bar{\epsilon}_{z,j}\right)^2}} \tag{7.6}$$

The selected regressor is only added to the model if it results in a significant improvement to the model itself. This is assessed through a F-statistic and is given by eq. 7.7 whereby F_0 denotes the test statistic and F_{IN} is the user-defined cutoff based on a significance level, α, of the test statistic (i.e. $F_{IN} = F(\alpha; 1, N-p-1)$) [71]. The choice of significance level is up to the designer. For aircraft systems, it is often the case that $N \gg 100$ and $p < 10$ (thus, $N >> p$) implying that the effect of p on F_{IN} is negligible. Therefore, F_{IN} is assumed to be constant and is commonly set to $F_{IN} = 4$ [71].

$$F_0 = \frac{SS_R(\hat{\boldsymbol{\theta}}_{p+j}) - SS_R(\hat{\boldsymbol{\theta}}_p)}{s^2} > F_{IN} \tag{7.7}$$

In eq. 7.7, $\hat{\boldsymbol{\theta}}_p$ gives the model parameters without the added regressor, $\boldsymbol{\xi}_j$, while $\hat{\boldsymbol{\theta}}_{p+j}$ represents the model parameters with the addition of $\boldsymbol{\xi}_j$. In both cases, the parameters can be estimated through OLS using eq. 7.3. $SS_R(\hat{\boldsymbol{\theta}})$ denotes the regression sum of squares and is defined in eq. 7.8 where N gives the number of measurement points and $\bar{z}$ the mean measurement value [71].

$$SS_R(\hat{\boldsymbol{\theta}}) = \hat{\boldsymbol{\theta}}^T \mathbf{A}^T \mathbf{z} - N\bar{z} \tag{7.8}$$

s^2 denotes the fit error variance and is defined by eq. 7.9 with p denoting the current number of regressors in the model [71].

$$s^2 \triangleq \hat{\sigma}^2 = \frac{\sum_{i=1}^{N} \left(z[i] - \mathbf{A}\hat{\boldsymbol{\theta}}\right)^2}{N-p-1} \tag{7.9}$$

If the condition $F_0 > F_{IN}$ is satisfied, then the regressor $\boldsymbol{\xi}_j$ is added to the model. It may be the case that the addition of $\boldsymbol{\xi}_j$ to the model makes some previously added regressor superfluous. Therefore, following the addition of a regressor $\boldsymbol{\xi}_j$, all model regressors are examined for redundancy [71]. It is important to note that, for this check, the regressors are not adjusted and therefore remain in their original form. For p model regressors, the quality of each regressor, $\boldsymbol{\xi}_k$, is evaluated through the test statistic defined in eq. 7.10 where $\hat{\boldsymbol{\theta}}_{p-k}$ denotes the parameters corresponding to a model with all p regressors except $\boldsymbol{\xi}_k$ [71].

$$F_{0,k} = \frac{SS_R(\hat{\boldsymbol{\theta}}_p) - SS_R(\hat{\boldsymbol{\theta}}_{p-k})}{s^2} \tag{7.10}$$

Should any of the regressors satisfy the condition $F_{0,k} < F_{OUT}$ where F_{OUT} is another user-specified threshold, then regressor with the lowest $F_{0,k}$ is removed from the model. As with F_{IN}, typically $F_{OUT} = 4$ [6, 71]. Should $\boldsymbol{\xi}_k = \boldsymbol{\xi}_j$ (i.e. should the removed regressor be the same as the just added regressor) then the algorithm terminates. If this is not the case, or if no regressors meet the condition $F_{0,k} < F_{OUT}$, then the algorithm continues adding regressors until the termination condition is met.

However, this termination condition does not consider the over-fitting of the model to the measurement data, z [6, 71]. Therefore, additional stopping conditions may be implemented to mitigate this effect. For example, the Predict Square Error (PSE) given by eq. 7.11 may be used, which prevents overfitting due to too many regressor terms [6, 71]. In eq. 7.11, $\hat{\mathbf{y}} = \mathbf{A}\hat{\boldsymbol{\theta}}$ represents the model predictions and σ^p_{max} is the a constant defined by eq. 7.12. The second term in eq. 7.11 represents the penalty for a model with p regressor terms and therefore scales with increasing p.

$$PSE \triangleq \frac{1}{N}(\mathbf{z}-\hat{\mathbf{y}})^T(\mathbf{z}-\hat{\mathbf{y}}) + \sigma^2_{max}\frac{p}{N} \tag{7.11}$$

$$\sigma^2_{max} = \frac{1}{N}\sum_{i=1}^{N}(z[i]-\bar{z})^2 \tag{7.12}$$

The PSE has previously been used by Sun et al. [6] as a stopping condition for the model structure selection of a high-speed quadrotor model. The PSE stopping condition is met once the PSE of the previous iteration is less than that of the current iteration (i.e. $PSE \geq PSE_{last}$). Having defined an approach for selecting an appropriate model structure based on a pool of model candidates, all that is left is to determine this candidate pool itself.

To demonstrate a potential method for defining model candidates, the selection process of Sun et al. [6, 7] will be used. However, regardless of the avenue taken to obtain the model regressors, the application of the stepwise regression algorithm is the same.

7.1.2 Selecting model candidates

A candidate pool of regressors can be determined through knowledge on the first principle models and observations from flight test data [6, 7].

The basic dynamics of the quadrotor can be captured by using the first principle model of the quadrotor (namely, eqs. 4.6 and 4.8 in section 4.2). By investigating the relations between state variables in the first principle model, it follows that polynomial functions may be used to describe these relations. For a polynomial of degree, d, with n independent variables, the number of terms, N_t, in the candidate pool excluding the constant term[†] (i.e. bias vector in A) is given by

$$N_t = \frac{(d+n)!}{n!d!} - 1 \tag{7.13}$$

Therefore, a polynomial with $d = 3$ and $n = 3$ has $N_t = 19$ terms. As an example, the candidate pool for a polynomial given by $P^3(u, |v|, w)$ is: $\{u$, $|v|$, w, u^2, $|v|^2$, w^2, $u|v|$, uw, $|v|w$, u^3, $|v|^3$, w^3, $u^2|v|$, u^2w, $|v|^2u$, $|v|^2w$, w^2u, $w^2|v|$, $u|v|w\}$.

As aforementioned, the simple first principle model fails to capture the relevant dynamics in the high-speed regime. Therefore, by using insights from chapter 5 (i.e. extensions to the simple quadrotor model), the polynomial candidate pool may be extended to capture these effects.

Force model candidates

Recall from chapter 5 that thrust variance and blade flapping are two aerodynamic effects commonly treated in quadrotor literature. Thrust variance influences the total thrust generated by the quadrotor while the blade flapping induces additional forces perpendicular to this thrust vector. Knowledge of these phenomena may be used to guide the choice of the force model candidates.

From section 5.3.1, the additional thrust (eq. 5.16) induced by the thrust variance effect was found to be

$$T_a = \kappa_2 V^2\cos^2(\alpha_r) + \kappa_3\left(V\sin(\alpha_r) + [v_i - v_h]\right)\sum_{j=1}^{4}\omega_j$$

[†]Note that the bias vector is excluded from the candidate pool since it is assumed to be present in the regressor matrix, A. Therefore, the candidate pool discussed here refers to only the potential regressors up for selection during stepwise regression, and not the fixed regressors (e.g. the bias vector).

Where T_a denotes the additional thrust, V represents the airspeed of the quadrotor, α_r the angle of attack, v_i the induced velocity, v_h the induced velocity during hover and ω_j gives the rotational speed of the j^{th} rotor ($j \in [1, \ldots, 4]$).

The total force, F_z, acting on the quadrotor along the z_B-axis is given by eq. 7.14 with T_h denoting the thrust during hover and D_z the drag induced by the aerodynamic effects.

$$F_z = -T + D_z = -(T_a + T_h) + D_z \tag{7.14}$$

Substituting eq. 5.16 into eq. 7.14 yields

$$F_z = -\{\kappa_2 V^2 \cos^2(\alpha_r) + \kappa_3 \left(V \sin(\alpha_r) + [v_i - v_h]\right) \sum_{i=j}^{4} \omega_j + \kappa_0 \sum_{j=1}^{4} \omega_j)\} + D_z \tag{7.15}$$

The relative velocity, V, of the quadrotor can be decomposed into components defined in the body frame (i.e. $[u, v, w]^T$). The velocity, w, is given by the projection of V onto the z_B-axis. In other words, $w = V \sin(\alpha_r)$. Similarly, u and v can be described by the projection of the relative velocity onto the x_B-y_B-plane, which is denoted by $u^2 + v^2 = V^2 \cos^2(\alpha_r)$. Therefore, the force model can be rewritten as

$$F_z = -\{\kappa_2(u^2 + v^2) + \kappa_3 \left(w + [v_i - v_h]\right) \sum_{j=1}^{4} \omega_j + \kappa_0 \sum_{j=1}^{4} \omega_j)\} + D_z \tag{7.16}$$

However, this force model fails to consider interaction effects between the rotors and the drone frame. Moreover, accurately determining $[v_i - v_h]$ and D_z can prove challenging, thereby limiting the practicability of this force model. Instead, Sun et al. [6] propose a polynomial model of F_z, based on eq. 7.16, which can account for these interaction effects and approximate $[v_i - v_h]$ and D_z.

To adequately capture interaction effects between the states, it follows that a polynomial model of F_z should have a minimum order of three due to the dependence of F_z on the states u, v, and w. A suitable polynomial model of F_z is given by eq. 7.17. Here, P^d denotes a polynomial of degree d. The first term in eq. 7.17 represents the thrust during hover, which can be easily be identified from flight data during hover. The second term, $P^3_{Fz,1}$, accounts for $\kappa_2(u^2 + v^2)$ and D_z among other aerodynamic effects and interactions. The last term, $P^3_{Fz,2}$, relates to $\kappa_3 \left(w + [v_i - v_h]\right)$.

$$F_z = -\kappa_0 \sum_{j=1}^{4} \omega_j^2 + P^3_{Fz,1}\left(u, |v|, w\right) + P^3_{Fz,2}\left(u, |v|, w\right) \sum_{j=1}^{4} \omega_j \tag{7.17}$$

With this, the candidate regressor pool for F_z is defined. The sign of the forward-backward movement (i.e. u) is conserved to investigate forward-aft rotor interactions since only forward flight is considered in [6]. Consequently, $|v|$ is used here to avoid issues with the sign of the lateral motion of the quadrotor, since the dynamics of these motions are hypothesized to be independent of the sign [6]. Note that in their later work, Sun et al. [7] take the absolute value of both u and v for their F_z model since they consider forward flight with varying heading angles.

Blade flapping induces significant aerodynamic drag, perpendicular to the thrust force, on the quadrotor during flight. In fact, these drag forces were found to be more influential than the pressure and parasitic drag of the quadrotor frame itself [6]. For this reason, the drag of the airframe is neglected in the polynomial model of Sun et al. [6]. Following the blade flapping induced force derivations of section 5.3.2, the forces acting along x_B and y_B can be described through eq. 7.18 [39]. Here, R denotes the radius of a rotor, A_{1_c} and A_{1_s} are constants which describe the blade flapping, and d_x and d_y are the induced drag coefficients for the x and y directions respectively.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \sum_{j=1}^{4} T_j \left(\frac{1}{\omega_j^B R} \begin{bmatrix} A_{1_c} & (-1)^{j-1} A_{1_s} \\ (-1)^j A_{1_s} & A_{1_c} \end{bmatrix} + \begin{bmatrix} d_x & 0 \\ 0 & d_y \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} \tag{7.18}$$

One assumption made in the derivation of the blade flapping effects is that the quadrotor propellers rotate at similar speeds [39]. Consequently, for a nominal quadrotor, the terms associated with A_{1_s} cancel out due to the counter rotation of the propellers yielding

$$F_x = u \sum_{j=1}^{4} T_j \left(\frac{A_{1_c}}{\omega_j R} + d_x \right) \tag{7.19}$$

$$F_y = v \sum_{j=1}^{4} T_j \left(\frac{A_{1_c}}{\omega_j R} + d_y \right) \tag{7.20}$$

It is evident that F_x and F_y also depend on the thrust produced by each rotor, T_j, and by extension are also influenced by the thrust variance effect. Therefore, as with deriving the model for F_z, the thrust can be decomposed into $T_j = T_{j,a} + T_{j,h}$. Therefore, eqs. 7.19 and 7.20 are equivalent to eqs. 7.21 and 7.22 respectively.

$$F_x = u\kappa_0 \sum_{j=1}^{4} \left(\frac{A_{1_c}}{R} \omega_j + d_x \omega_j^2 \right) + u \sum_{j=1}^{4} \left[\left(\kappa_2 (u^2 + v^2) + \kappa_3 (w + v_a) \omega_j \right) \left(\frac{A_{1_c}}{\omega_j R} + d_x \right) \right] \tag{7.21}$$

$$F_y = v\kappa_0 \sum_{j=1}^{4} \left(\frac{A_{1_c}}{R} \omega_j + d_x \omega_j^2 \right) + v \sum_{j=1}^{4} \left[\left(\kappa_2 (u^2 + v^2) + \kappa_3 (w + v_a) \omega_j \right) \left(\frac{A_{1_c}}{\omega_j R} + d_y \right) \right] \tag{7.22}$$

By looking at the first terms of eqs. 7.21 and 7.22 respectively, it is evident that there is a dependence of the state on both ω_j and ω_j^2. Intuitively, there is a strong collinearity between these two variables for the majority of the quadrotor flight envelope. Such high correlations between the regressors deteriorates the performance of the resultant model. Therefore, to ameliorate this, Sun et al. [6] propose that ω_j^2 be absorbed by ω_j such that eqs. 7.21 and 7.22 become eqs. 7.23 and 7.24 respectively.

$$F_x = u\kappa_0 \sum_{j=1}^{4} \omega_j \left(\frac{A_{1_c}}{R} + d_x \right) + u \sum_{j=1}^{4} \left[\left(\kappa_2 (u^2 + v^2) + \kappa_3 (w + v_a) \omega_j \right) \left(\frac{A_{1_c}}{\omega_j R} + d_x \right) \right] \tag{7.23}$$

$$F_y = v\kappa_0 \sum_{j=1}^{4} \omega_j \left(\frac{A_{1_c}}{R} + d_x \right) + v \sum_{j=1}^{4} \left[\left(\kappa_2 (u^2 + v^2) + \kappa_3 (w + v_a) \omega_j \right) \left(\frac{A_{1_c}}{\omega_j R} + d_y \right) \right] \tag{7.24}$$

Notice that the second term of both eqs. 7.23 and 7.24 resembles the first term of the thrust model (see eq. 7.16). Therefore, the second terms of the drag models may also be approximated by a polynomial similar to that of eq. 7.17. Doing so also harbors the advantage of accounting for interaction effects, which are otherwise neglected in eqs. 7.23 and 7.24. Therefore, a suitable polynomial drag and thrust model can be described by eq. 7.25 where X_1 and Y_1 are parameters which can be identified from flight data [6].

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} X_1 u \sum_{j=1}^{4} \omega_j + P_{Fx1}^3 (u, |v|, w) + P_{Fx2}^3 (u, |v|, w) \sum_{j=1}^{4} \omega_j \\ Y_1 v \sum_{j=1}^{4} \omega_j + P_{Fy1}^3 (|u|, v, w) + P_{Fy2}^3 (|u|, v, w) \sum_{j=1}^{4} \omega_j \\ -\kappa_0 \sum_{j=1}^{4} \omega_j^2 + P_{Fz,1}^3 (u, |v|, w) + P_{Fz,2}^3 (u, |v|, w) \sum_{j=1}^{4} \omega_j \end{bmatrix} \tag{7.25}$$

Let $\mathbf{F}_a = [F_{a,x}, F_{a,y}, F_{a,z}]^T$ denote the components of the force model which contain the candidate regressors. That is, let:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} X_1 u \sum_{j=1}^{4} \omega_j + F_{a,x} \\ Y_1 v \sum_{j=1}^{4} \omega_j + F_{a,y} \\ -\kappa_0 \sum_{j=1}^{4} \omega_j^2 + F_{a,z} \end{bmatrix} \tag{7.26}$$

By applying the aforementioned stepwise regression algorithm, Sun et al. [6] obtained eq. 7.27 to describe $F_{a,x}$ and $F_{a,z}$. Note that a model structure for $F_{a,y}$ was not presented in [6] since only forward flight was being considered and presumably due to the similarity in structure to $F_{a,x}$. However, to avoid speculation, no $F_{a,y}$ model is presented here either since the measurement data used for identification was not readily available.

$$\begin{bmatrix} F_{a,x} \\ F_{a,z} \end{bmatrix} = \begin{bmatrix} X_2|v|w + X_3 u|v| + X_4 u w^2 \sum_{j=1}^{4} \omega_j + X_5 u w \sum_{j=1}^{4} \omega_j + X_6 u w^2 + X_7 u |v|^2 + X_8 u |v| \\ Z_0 + Z_1 w \sum_{j=1}^{4} \omega_j + Z_2 u^2 + Z_3 |v|^2 + Z_4 u^2 w + Z_5 |v| w^2 + Z_6 u^2 \sum_{j=1}^{4} \omega_j \end{bmatrix} \tag{7.27}$$

Aside from Z_0, the order of the terms in eq. 7.27 reflects the order in which they were selected by the stepwise regression algorithm. Intuitively, some of the most influential terms of the drag model are related to the horizontal velocities. Likewise, the most influential term for the thrust model is related to the rotational speeds of the rotors. The expected dependence of $F_{a,z}$ on the horizontal velocities (see the first term of eq. 7.16) can also be seen. However, it should be noted that the order of regressor selection is sensitive to measurement data used for regression and, with slightly different measurement data, the regressor priorities may be different. Hence, the selected regressors should also be evaluated for their sensitivity in terms of priority before any conclusions may be drawn about their general significance to the model.

According to the results of Sun et al. [6], the identified force models outperform the simple quadrotor model. In hover conditions, where the simple model is valid, the identified force models emulate the prediction accuracy of the simple model. In the high-speed regime, the identified force models were able to accurately model the quadrotor dynamics while the simple quadrotor model fails to do so.

Moment model candidates

Unlike the derivation of the force model candidates, literature seldom considers (aerodynamic) moment models. Therefore, the model candidates can be formulated from observations on flight test data and physical insights on the underlying dynamics of the system. Sun et al. adopted this approach in [6] in order to identify an accurate pitching and yawing moment model of the quadrotor in the forward flight high-speed regime. The pitching moment is relevant for fast forward flight while the yawing manoeuvre is common in quadrotor videography and is prone to actuator saturation.

It is apparent that the thrust force produced by each rotor directly influences the resultant control moment. Therefore, the thrust variance effect also induces moment variations on the quadrotor [6]. This suggests that the difference in thrust between the front and aft rotors (i.e. U_q) may be an influential pitching moment model term. Likewise, the difference between the clockwise and counterclockwise rotor torques, U_r, should be considered for the yawing moment model. Following the definition of the quadrotor in this paper (i.e. fig. 4.6), U_q and U_r can be defined by eq. 7.28 and eq. 7.29 respectively.

$$U_q = \omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2 \tag{7.28}$$

$$U_r = -\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2 \tag{7.29}$$

Another effect influencing the moment models is aerodynamic damping [22]. The magnitude of damping is related to the rotational rate of the quadrotor, $\boldsymbol{\Omega}$. Subsequently, the pitch rate, q, should be added as a state for the pitching moment model and the yaw rate, r, for the yawing moment model [6, 7]. Additional moments affecting both moment models may originate from the interaction effects between rotors and the airframe along with moments from the airframe itself. Therefore, preliminary models of the pitching and yawing moments may be described by eq. 7.30 and eq. 7.31 respectively.

$$M_y = \ell\kappa_0 U_q + P_1^3(u, w, q)U_q + \varepsilon_{M,y} \tag{7.30}$$

$$M_z = \tau_0 U_r + P_1^3(u, v, r)U_r + \varepsilon_{M,z} \tag{7.31}$$

Here, ε_M denotes the unmodelled moments acting on the quadrotor. The first terms in these equations represent the control moments stemming from the simple quadrotor model while the second terms address the moments emanating from the thrust variance effect. Note that the constituents of velocity acting in-line with the moments are excluded from the state variables since their contribution to the moment is negligible.

Recall from the discussion pertaining to blade flapping in chapter 5 (section 5.3.2) that this aeordynamic effect also induces moments on the quadrotor. Namely, a pitching moment due to the tilting of the thrust vector (longitudinal moment) and another due to the flapping of the blades around the rotor hub (hub moment) [6]. The longitudinal moment is a function of the steady-state blade flapping angle and the rotor thrust. Therefore, the pitching moment is influenced by the interaction between the rotor angular rates, ω_j, and the velocity of the quadrotor. The hub moment depends on the stiffness of the propeller blades and the steady-state blade flapping angle, which itself is a function of the velocity. Therefore, eq. 7.30 can be expanded to eq. 7.32 to accommodate these effects.

$$M_y = \ell\kappa_0 U_q + P_1^3(u, w, q)U_q + P_2^3(u, w, q)\sum_{j=1}^{4}\omega_j + P_3^3(u, w, q) + \varepsilon_{M,y} \tag{7.32}$$

It is suspected that the aerodynamic interactions during high speed flight may diminish the thrust of the aft rotors, thereby leading to pitch up moments [7]. The reason being that, as the quadrotor is in forward flight, and depending on its angle of attack, the rear rotors may be caught in the wake of the front rotors. For a quadrotor in forward flight, the angle of attack is dependent on its velocity and is therefore already captured by the term $P_1^3(u, w, q)$ in eq. 7.32. Therefore, a suitable pitching moment model can be defined as eq. 7.33 [6].

$$M_y = \ell\kappa_0 U_q + P_1^3(u, w, q)U_q + P_2^3(u, w, q)\sum_{j=1}^{4}\omega_j + P_3^3(u, w, q) \tag{7.33}$$

The yawing moment model can be extended using two commonly modelled effects in literature, which are prevalent even in hover conditions. The first is the rotor inertia and gyroscopic moment previously discussed in chapter 5 (section 5.1). Another is the linear yaw damping term, γr, with damping coefficient γ [22]. Incorporating these effects yields eq. 7.34. The constants τ_0, $I_{r_{zz}}$, and γ can be estimated from measurements taken during hover [6].

$$M_z = \tau_0 U_r + I_{r_{zz}}\sum_{j=1}^{4}(-1)^j\dot{\omega}_j + \gamma r + P_1^3(u, v, r)U_r + \varepsilon_{M,z} \tag{7.34}$$

Sun et al. [6] postulate that the translational speed and rotation of the quadrotor influences its yawing moment and therefore propose eq. 7.35 as a polynomial model of the yawing moment.

The addition of P_2^3 is to address the potential variation in yawing moment as a function of the thrust while the P_3^3 polynomial captures general effects related solely to the airspeed and rotational rate of the quadrotor, such as the moments induced by the airframe.

$$M_z = \tau_0 U_r + I_{r_{zz}} \sum_{j=1}^{4} (-1)^j \dot{\omega}_j + \gamma r + P_1^3(u,v,r)U_r + P_2^3(u,v,r) \sum_{j=1}^{4} \omega_j + P_3^3(u,v,r) \tag{7.35}$$

Using the polynomial terms of eq. 7.33 and eq. 7.35 as the model candidates for the pitching and yawing moment models respectively, Sun et al. [6] identify the following model structures

$$\begin{aligned} M_y \;\; = \;\; & \ell\kappa_0 U_q + M_0 + M_1 u U_q + M_2 u^2 U_q + M_3 w U_q + M_4 u + M_5 u^3 + M_6 w \\ & + M_7 u w U_q + M_8 q + M_9 u \textstyle\sum_{j=1}^{4} \omega_j + M_{10} u^2 \sum_{j=1}^{4} \omega_j + M_{11} u w \end{aligned} \tag{7.36}$$

$$M_z = \tau_0 U_r + I_{r_{zz}} \sum_{j=1}^{4} (-1)^j \dot{\omega}_j + \gamma r + N_0 + N_1 v + N_2 u U_r + N_3 u^3 U_r + N_4 v^2 U_r + N_5 u v \tag{7.37}$$

From the selected model candidates in eq. 7.36 and eq. 7.37 it is clear that the thrust variance effect is contributes significantly to both the pitching and yawing moments. The moment generated by the motor itself, i.e., terms with $\sum_{j=1}^{4} \omega_j$ show little priority in the pitching moment model and are absent entirely from the yawing moment model [6]. For the pitching moment, this is likely due to the small offset between the rotor plane and the c.g. of the quadrotor (recall that the longitudinal moment depends on the distance to the c.g.). With regards to the yawing moment, influence of the rotor generated moments are likely captured sufficiently by the first and second terms of eq. 7.37.

These identified moment models (i.e. eq. 7.36 and eq. 7.37) were found to outperform the simple quadrotor model [6]. As with the force models, the moment models mirror the prediction performance of the simple quadrotor model during hovering conditions while the identified moment models are more accurate in the high-speed regime.

It should be noted that the prevalence of U_q and U_r in the moment models imply that these models are sensitive to the angle of sideslip of the quadrotor. Depending on the angle of sideslip, the aft rotors may be contained or free of the wake of the front rotors thereby affecting resultant moments. In fact, in a more recent high-speed quadrotor identification study by Sun et al. [7], the selected model regressors were found to differ depending on the sideslip angle. In this study, Sun et al. [7] utilized the same stepwise regression algorithm to identify the high-speed models but with different regressors. In [7], Sun et al. utilized the advance ratios (see eq. 7.38) to build the model regressors. This choice was motivated by the impracticality of modelling the rotors individually, since the sensor measurements of the quadrotor observe their combined effect. Therefore, a dimensionless approach is taken which scales the forces and moments by the average rotor speed [7]. As with their prior work in [6], Sun et al. in [7] found significant improvements in model accuracy when using the identified models over the simple quadrotor model.

$$\mu_x = \frac{u}{\sqrt{\frac{\sum_{j=1}^{4} \omega_j^2}{4}} R}, \quad \mu_y = \frac{v}{\sqrt{\frac{\sum_{j=1}^{4} \omega_j^2}{4}} R}, \quad \mu_z = \frac{w}{\sqrt{\frac{\sum_{j=1}^{4} \omega_j^2}{4}} R} \tag{7.38}$$

The improved performance of both variants of the high-speed models identified by Sun et al. [6, 7] exemplify the flexibility of the stepwise regression technique. The merits of using the system states [6] or their dimensionless variants [7] as a basis for the regressors are that the selected regressors can be easily interpreted as physical phenomena, contributing to understanding on

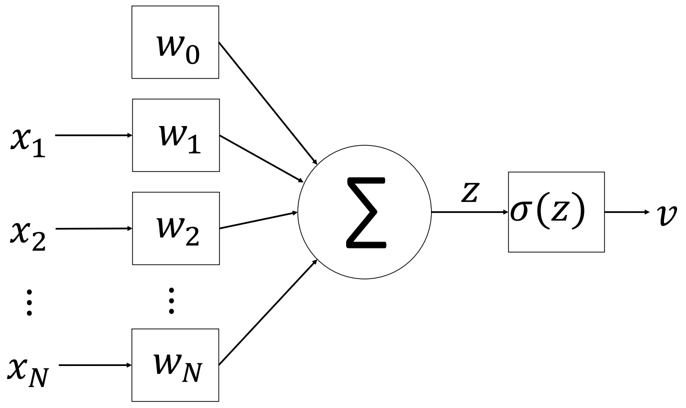


Figure 7.1: Illustration of the artificial neuron. The neuron receives inputs, $x_i, i \in [1, 2, \ldots, N]$, which are scaled by the weights, w_i of the neuron with w_0 denoting the bias term. The weighted sum of these inputs, z, passes through some activation function, $\sigma(z)$, resulting in output, v.

quadrotor dynamics in the high-speed regime. However, this dependence on the chosen model candidate pools also exposes a limitation of the stepwise regression technique. Namely, that a poor selection of candidate regressors - which may be due to a lack of understanding on the dynamics governing a system - will likely lead to a poor model. Therefore, stepwise regression may be unsuitable for systems for which little knowledge is available. Moreover, these methods require that the system is linear in the parameters, which may not always be the case.

7.2 Artificial Neural Networks

As is evident from the previous sections, knowledge on the quadrotor models is rather incomplete and irregular. This limits, to a certain extent, the use of gray-box models. Therefore, when investigating uncharted regions of the flight envelope, black-box system identification approaches are often employed. Artificial Neural Networks (ANNs) have emerged in recent decades as a particularly popular choice for such black-box modelling due to advancements in computational efficiency [73] and improvements to ANNs themselves, such as the development of unsupervised trainable deep ANNs (DNNs) [74].

ANNs are capable of capturing unknown non-linearities [75, 76] in a system and are thought to be universal function approximators [35]. The latter property is especially powerful since it facilitates the modelling of unobserved states from the measurement data, making ANNs particularly attractive as a system identification technique. It is no surprise, then, that ANNs have successfully been applied to identify models of aircraft [77, 78, 79], rotorcraft [80, 81], and, more recently, quadrotors [34, 35, 82, 83].

7.2.1 Mathematical foundations behind artificial neural networks

A brief introduction to the theoretical foundations behind the components and potential structures of an ANN are provided here. Through this, some considerations for constructing ANNs in a system identification framework may be highlighted.

The artificial neuron

An ANN is a system of interconnected artificial neurons designed to emulate biological neural networks. Reminiscent of a biological neuron, fig. 7.1 illustrates the structure of an artificial neuron which parameterizes its inputs into a single (nonlinear) output.

Let x_i denote an input to the neuron with $i \in [1, 2, \ldots, N]$ for N inputs. Each input is subsequently scaled by a corresponding adaptive weight, w_i. Included in the model of the neuron is a bias term denoted by w_0. The purpose of this bias term is analogous to the role of a constant in a polynomial function. The sum of these weighted inputs and bias term, z (see eq. 7.39), is subsequently transformed by some activation function, $\sigma(z)$, culminating in the neuron output, v.

$$z = w_0 + \sum_{i=1}^{N} w_i x_i \tag{7.39}$$

The choice of activation function is up to the designer of the NN and depends on the desired neuron output and position of the neuron in the NN structure. Some commonly used activation functions are summarized in table 7.1. In general, activation functions can be classified as either projection functions or as kernel functions. The distinction being that kernel functions, such as the radial basis function (RBF), are centered around some fixed point[†], c, and asymptotically approach zero in directions radiating away from this point. Therefore, kernel functions transform the inputs to a local region of the model. Conversely, inputs fed through projection functions have a global influence on the model. Naturally, projection activation functions are suitable to model global model trends while kernel activation functions are equipped to capture local dynamics. Note, however, that an ANN using RBFs cannot be trained locally. Thus, when a local change occurs in the system (e.g. due to an actuator fault), the *entire* network needs to be retrained to accommodate this change.

Interconnected neurons: the neural network

The structured organization of neurons, with the same or different activation functions, defines the network. The most simple NN architecture is perhaps the feed-forward neural network, illustrated in fig. 7.2, wherein information only flows from n inputs to m outputs. A general feed-forward neural network is composed of an input layer, K hidden layers, and an output layer. It should be noted that the input layer does not contain neurons and instead relays the inputs to the neurons of the first hidden layer.

Another commonly used NN architecture is the recurrent neural network (RNN), which is specialized for time invariant systems. Consequently, RNNs are particularly alluring for system identification and control. In fact, Mohajerin et al. [83] applied an RNN in a hybrid configuration with first principle techniques to identify a model of a quadrotor. RNNs exploit the temporal invariance of the system to reduce the number of parameters (namely, weights) in the network. Figure 7.3 illustrates the RNN concept wherein the input-output mapping, while dependent on previous states, is invariant of the time. Since RNNs learn in a similar way to feed-forward ANNs, the rest of this section uses the simpler feed-foward ANN to facilitate explanations.

The structure of a feed-foward ANN is determined by the number hidden layers, K, and the number of neurons in a hidden layer, n_k. These variables considerably impact the resultant performance of the ANN. The simplicity in improving model performance is an integral factor behind the popularity of ANNs in modelling. George Cybenko famously postulated theorem 1 [84] implying that the predictive capabilities of ANN models can be improved by simply adding more neurons to a given layer.

Theorem 1. *Provided that there are no constraints imposed on the available number of neurons or the magnitudes of weights, then a single hidden layer feedforward neural network with sigmoid activation functions can approximate continuous functions with arbitrary precision.*

[†]Typically, the origin is taken as the fixed point i.e. $c = 0$, for simplicity.

Activation function	Classification	Equation	Comments
Sigmoid	Projection function	$\sigma(z) = \frac{1}{1+e^{-z}}$	+ Output bounded to $[0, 1]$ – Vanishing gradient – Not zero-centered
Tanh	Projection function	$\sigma(z) = \frac{2}{1+e^{-2z}} - 1$	+ Output bounded to $[-1, 1]$ + Zero-centered – Vanishing gradient
ReLU	Projection function	$\sigma(z) = \begin{cases} 0 & \text{if} \quad z < 0 \\ z & \text{if} \quad z \geq 0 \end{cases}$	+ More efficient learning + Simple – Dying ReLU problem
Leaky ReLU	Projection function	$\sigma(z) = \begin{cases} \alpha(e^z - 1) & \text{if} \quad z \leq 0 \\ z & \text{if} \quad z > 0 \end{cases}$	+ Solves dying ReLU problem – Additional parameter α – Sometimes inconsistent results
Radial Basis Functions	Kernel function	$\sigma(z) = e^{-z^2}$	+ Local modelling + Straight forward optimization – Computationally expensive

Table 7.1: Summary of commonly used neural network activation functions. Some of the advantages (+) and disadvantages (-) of each function are highlighted under the 'Comments' column.

In fact, Andrew Barron [85] quantified the integrated squared error, J, of a feed-foward NN with a single hidden layer, containing n_k neurons, using sigmoidal activation functions as eq. 7.40. Interestingly, this is independent of the input dimension space, p. Compare this to the integrated squared error for basis function expansions (e.g. polynomials) with n_k terms (i.e. regressors) which is given by eq. 7.41. The work of Cybenko [84] and Barron [85] clearly encourage the use of ANNs due to their desirable approximation capabilities.

$$J = \mathcal{O}\left(\frac{1}{n_k}\right) \tag{7.40}$$

$$J = \mathcal{O}\left(\frac{1}{n_k^{2/p}}\right) \tag{7.41}$$

It should be highlighted that, although ANNs are suitable for high-dimensional problems, eq. 7.40 only provides a theoretical limit and does not reflect the practical accuracies of an ANN. In fact, blindly adding neurons leads to diminishing returns. Increasing the number of neurons increases the susceptibility of the feed-forward ANN to over-fitting. Such wide ANNs are adept at memorization and begin to model the noise in the data instead of the dynamics behind the data. More neurons also entails more trainable network parameters, demanding

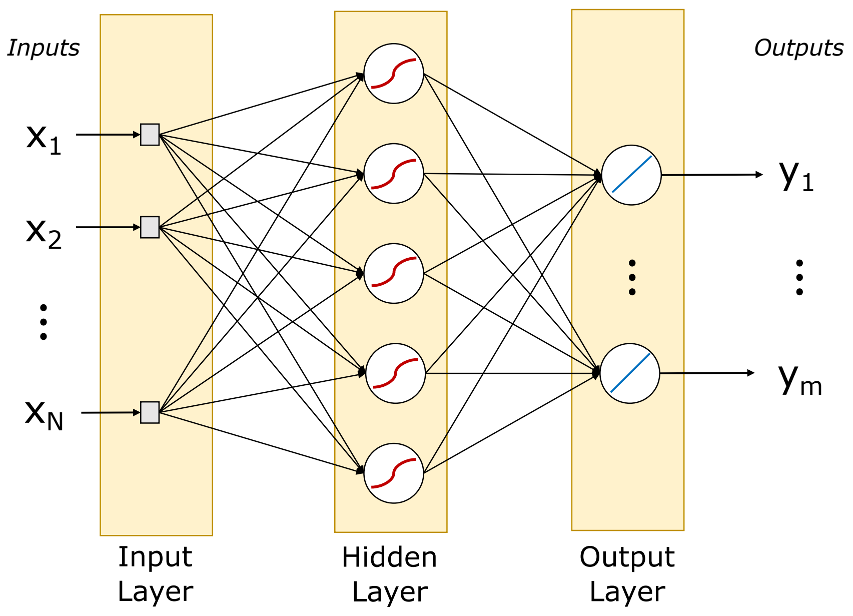


Figure 7.2: Network architecture of a single hidden layer feed-forward artificial neural network. Neurons are indicated through circular blocks while the input pass-through nodes are given by square blocks.

more computational effort. Using fewer neurons relieves this computational load. However, too few neurons may result in under-fitting whereby the feed-foward ANN fails to adequately capture the system dynamics. Strategies to balance these are discussed later in section 7.2.2.

An alternative to manipulating the number of neurons is to instead change the number of hidden layers. In general, increasing the number of hidden layers improves the abstraction capabilities of the ANN. For example, an ANN with no hidden layers is only suitable for linear separable functions [86]. With one hidden layer the ANN can now represent functions with a continuous mapping from $\mathbb{R}^n \rightarrow \mathbb{R}^m$. Accordingly, single hidden layer feed-forward ANNs are suitable for most modelling applications [86]. Increasing the number of hidden layers even further enables the network to learn intermediate features of the dataset. Consider, for example, the application of DNNs to image classification. By visualizing various hidden layer outputs of a convolutional NN (CNN), Zeiler et al. [73] were able to uncover that deeper layers extract increasingly complex features, derived from the shallower layers. For instance, the first hidden layer appears to capture elementary features of the image such as edges; the following layer extracts more complex features derived from these elementary features, such as textures; subsequent layers show more class-specific features such as faces or legs and subsequently vary more across inputs in comparison to shallower layers [73]. Therefore, unlike increasing the number of neurons, increasing the number of hidden layers facilitates generalization through feature extractation instead of memorization. This property of DNNs explains their ever-growing popularity and preference over wide ANNs. This is not to say, however, that adding layers resolves the issue of over-fitting. In fact, over-fitting is still a considerable issue for DNNs. Additionally, as with wide ANNs, the computational demand of training scales with the number of hidden layers. This is exemplified in the work of Mohajerin et al. [83] wherein the training of the deep RNN for quadrotor system identification was facilitated by a powerful GPU (GeForce Titan Xp).

Training through backpropagation

For ANNs, training is accomplished through a learning procedure known as 'backpropagation' [87, 88, 89, 90] which aims to find a set of network weights that minimize the error between the

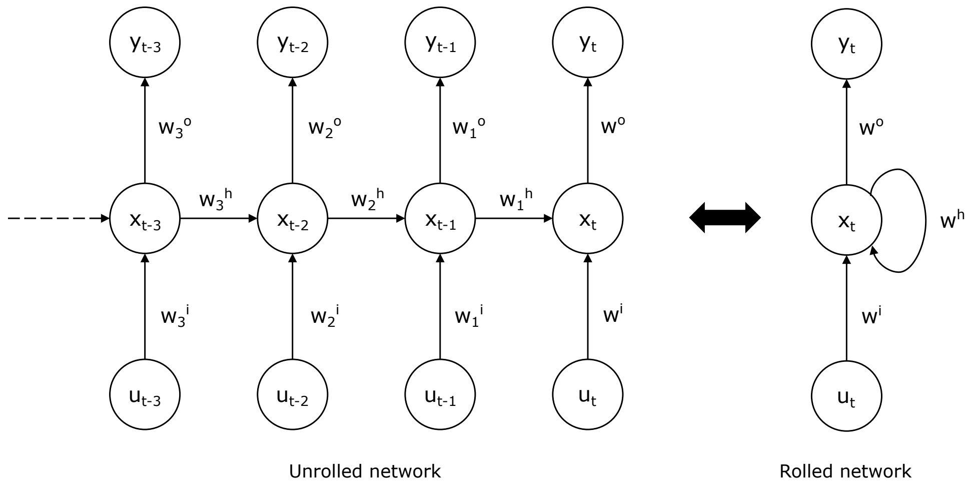


Figure 7.3: Illustration of a Recurrent Neural Network (RNN). An RNN exploits the temporal invariance of a process to reduce the number of network parameters. In this case, the weights can be simplified following $w^3 = w^2 = w^1 = w$. Shown are the input, hidden and output weights denoted by i,h, and o respectively. The inputs to the system are given by u, the state of the system is represented by x while y denotes the output. Here, t denotes the time.

network outputs and desired outputs, for all (especially unseen) inputs. The backpropagation procedure exploits the error between the network output(s) and its target value(s) to update the network weights. This is achieved by working backwards, from the output to the inputs, to recursively obtain the gradient of the error with respect to the network weights. Subsequently, the weights can be adapted to minimize this error through gradient descent.

Consider a feed-forward neural network with an input layer, an arbitrary amount of hidden layers, and an output layer. For this network, impose the constraints that there are no connections within a given layer or from one layer to previous layers and that the network is dense[†] [90]. Let $\mathbf{x}$ denote the inputs, $\mathbf{y}$ the *network* outputs, $\mathbf{t}$ the targets (i.e. $\mathbf{y} = \hat{\mathbf{t}}$), and w_{ij}^k the weight from neuron i in layer $k-1$ to neuron j in layer k. The error between the network outputs and targets is defined by some user-set cost function, $J(\mathbf{x}, \mathbf{w})$. The squared error (eq. 7.42) is frequently used as the cost function in backpropagation [87, 90, 91]. Nevertheless other functions may also be used so long as they produce informative error gradients which correspond to the desired convergence characteristics of the network (e.g. towards the mean of the dataset or the median).

$$J(\mathbf{x}, \mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{h=1}^{m}(t_{n,h} - y_{n,h})^2 \tag{7.42}$$

The backpropagation learning algorithm is initialized with arbitrary network weights, from which the corresponding outputs of the network are derived [87]. Subsequently, the error between the network outputs and targets may be computed using eq. 7.42.

Starting from an arbitrary network output, y_h, the error gradient with respect to the weights can expressed using the chain rule through eq. 7.43 where z_h gives the weighted sum of the inputs (including the bias term) into output neuron h in the output layer K (i.e. z_h is given by eq. 7.44) [87, 90]. Note that y_h is used as an example here but, in principle, the process is equivalent for the other outputs $h \in [1, \ldots, m]$. The total error is then obtained by summing over these

[†]For NNs, the term 'dense' means that the layers are fully connected; i.e. a neuron in layer k receives inputs from all neurons in the previous layer, $k-1$, and is connected to all neurons in layer $k+1$

outputs.

$$\frac{\partial J}{\partial w_{ih}} = \frac{\partial J}{\partial z_h}\frac{\partial z_h}{\partial w_{ih}} \tag{7.43}$$

$$z_h = \sum_{i=0}^{r^{K-1}} v_i^{K-1} w_{ih}^K \tag{7.44}$$

Using eq. 7.42, $\frac{\partial J}{\partial z_h}$ can be resolved for some arbitrary (differentiable) activation function, σ, through

$$\begin{aligned} \frac{\partial J}{\partial z_h} &= \frac{\partial}{\partial z_h}\left(\frac{1}{2}\sum_{n=1}^{N}(t_n - \sigma(z_{n,h}))^2\right) \\ &= \sum_{n=1}^{N}(t_n - \sigma(z_{n,h}))\sigma'(z_{n,h}) \end{aligned} \tag{7.45}$$

Likewise, $\frac{\partial z_h}{\partial w_{ih}}$ can be readily obtained using eq. 7.44 yielding eq. 7.46. Essentially, eq. 7.46 indicates that the influence of weight, w_{ih}, between neuron i and neuron h depends only on the output of neuron i, v_i^{K-1}. This makes intuitive sense since the weight w_{ih} connects neuron i with neuron h and is therefore invariant of other neuron outputs from the same layer.

$$\frac{\partial z_h}{\partial w_{ih}} = v_i^{K-1} \tag{7.46}$$

Substituting eq. 7.45 and eq. 7.46 into eq. 7.43 leads to

$$\frac{\partial J}{\partial w_{ih}} = v_i^{K-1}\sum_{n=1}^{N}(t_n - \sigma(z_{n,h}))\sigma'(z_{n,h}) \tag{7.47}$$

However, the output of neuron i also depends on the outputs, and therefore weights, of the preceding neurons. Fortunately, the outputs of one neuron depend linearly on the inputs to the neuron and weights thereof [90]. Consequently, the same procedure for deriving eq. 7.47 may be applied per layer of the ANN. For this reason, adding neurons and layers increases the training time considerably as more paths are added to the backpropagation algorithm.

For an arbitrary neuron, j, in an arbitrary layer, $1 \leq k < K$, the weighted sum of inputs, z_j^k, can be expressed through eq. 7.48 wherein v_l^{k-1} denotes the output from neuron l in layer $k-1$. w_{lj}^k describes the weights of the respective connections. The number of outputs from the layer $k-1$ is given by r^{k-1}.

$$z_j^k = \sum_{l=0}^{r^{k-1}} v_l^{k-1} w_{lj}^k \tag{7.48}$$

In a similar fashion to eq. 7.43, the partial derivative of the cost function with respect to the weights can be written as

$$\frac{\partial J}{\partial w_{ij}^k} = \frac{\partial J}{\partial z_j^k}\frac{\partial z_j^k}{\partial w_{ij}^k} \tag{7.49}$$

$\frac{\partial J}{\partial z_j^k}$ can be expressed as

$$\frac{\partial J}{\partial z_j^k} = \sum_{l=1}^{r^{k+1}} \frac{\partial J}{\partial z_l^{k+1}}\frac{\partial z_l^{k+1}}{\partial z_j^k} \tag{7.50}$$

Here, the key difference with the prior derivation for the output layer is that there are now r^{k+1} outputs (due to the dense nature of the hidden layers) which all have their own respective 'errors' that have been backpropagated from the outputs of the network. Per definition, z_l^{k+1} is related to z_j^k through eq. 7.51 with σ the activation function of the hidden layers.

$$z_l^{k+1} = \sum_{j=1}^{r^k} w_{lj}^{k+1} \sigma(z_j^k) \tag{7.51}$$

Let

$$q_l^{k+1} = \frac{\partial J}{\partial z_l^{k+1}} \tag{7.52}$$

thus eq. 7.50 can be expressed as

$$\frac{\partial J}{\partial z_j^k} = \sigma'(z_j^k) \sum_{l=1}^{r^{k+1}} q_l^{k+1} w_{lj}^{k+1} \tag{7.53}$$

As with the output layer, $\frac{\partial z_j^k}{\partial w_{ij}^k}$ can be obtained using eq. 7.48 to yield

$$\begin{aligned} \frac{\partial z_j^k}{\partial w_{ij}^k} &= \frac{\partial}{\partial w_{ij}^k} \left(\sum_{l=0}^{r^{k-1}} v_l^k w_{ij}^k \right) \\ &= v_i^{k-1} \end{aligned} \tag{7.54}$$

Therefore, the backpropagation formula for an arbitrary neuron in the network can be found through eq. 7.55. Using this equation, the influence of each of the weights of the network on the cost function can be determined and subsequently updated in the appropriate direction.

$$\frac{\partial J}{\partial w_{ij}^k} = \sigma'(z_j^k) v_i^{k-1} \sum_{l=1}^{r^{k+1}} q_l^{k+1} w_{lj}^{k+1} \tag{7.55}$$

The question now is *how* to adapt these weights? Since the gradient gives an indication of the direction in which the weights should be changed, eq. 7.56 may be used to update the weights proportionally to the gradient through a constant, α [87, 90].

$$\mathbf{w}_{new} = \mathbf{w}_{old} - \alpha \nabla J(\mathbf{w}_{old}) \tag{7.56}$$

Where $\nabla J(\mathbf{w}_{old})$ denotes the gradient (expanded in eq. 7.57) of all the weights, $w_r \in [1, R]$, in the network.

$$\nabla J(\mathbf{w}_{old}) = \frac{\partial J}{\partial \mathbf{w}_{old}} = \left[\frac{\partial J}{\partial w_{old,1}}, \frac{\partial J}{\partial w_{old,2}}, \ldots, \frac{\partial J}{\partial w_{old,R}} \right]^T \tag{7.57}$$

For a database of size, N, the gradient may be computed as

$$\nabla J(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} \left(\frac{\partial J(\mathbf{t}_n, \mathbf{x}_n, \mathbf{w})}{\partial \mathbf{w}} \right) \tag{7.58}$$

Taking eq. 7.58 with the weight update rule (eq. 7.56) results in the so-called gradient descent method. For large N, using the gradient descent may be inefficient. Therefore, the database can be split into batches of size, $n << N$, containing data points (i.e. $t_i; x_i$) that are independent of each other and identically distributed. This is known as the stochastic gradient descent, where the gradient may be estimated by eq. 7.59.

$$\hat{\nabla} J(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\partial J(\mathbf{t}_i, \mathbf{x}_i, \mathbf{w})}{\partial \mathbf{w}} \right) \tag{7.59}$$

For some applications, the simple (stochastic) gradient descent may be inefficient (e.g. too slow). In light of this, some extensions to this method are employed in literature to improve the convergence performance. For instance, the velocity of the gradient descent may be used to allow the algorithm to take larger update steps if the gradient is consistently along the same direction. This is known as stochastic gradient descent with momentum, and is defined in eq. 7.60 where i denotes current update iteration and $\gamma \in [0, 1]$ is a constant representing how influential previous weight updates are [90].

$$\begin{aligned} \Delta \mathbf{w}_{new}[i] &= -\alpha \nabla J(\mathbf{w}_{old}[i]) + \gamma \Delta \mathbf{w}_{new}[i-1] \\ \mathbf{w}_{new}[i] &= \mathbf{w}_{old}[i] - \Delta \mathbf{w}_{new}[i] \end{aligned} \tag{7.60}$$

However, too large of a step can result in poor convergence to the minima of the cost function. For instance, the algorithm may be unable to venture into a valley since the step size is wider than this valley. Hence, other methods, such as RMSProp [92], aim to reduce the update step size over time to mitigate this. Better yet, the principles of momentum and decreasing the step size over time can be combined. One such method is ADAM proposed by Kingma and Ba [92]. The choice of the optimizer depends on the requirements imposed on the network (e.g. convergence time and accuracy).

7.2.2 Considerations for training artificial neural networks

There are several design choices to be made when constructing an ANN, many of which influence its subsequent performance. A small, yet well-defined, ANN could easily outperform a complex, but poorly designed, one. Hence, some considerations to guide ANN design choices are presented here.

Hyperparameter choice

As is perhaps evident from prior discussions, there are several different parameters in an ANN which can be tuned. These are known as hyperparameters and are traditionally set before training. Much of the challenges in designing a good ANN model resides in the appropriate selection of these hyperparameters.

The hyperparameters of the neuron itself include the activation function and weight initialization. The choice of activation function (see table 7.1 for some examples) relates to the desired output and training characteristics of the neuron in question. For example, the sigmoid function is typically employed in the output layer and is used for binary classifications due to the bounds of its output (i.e. $[0, 1]$). However, the sigmoid function, and similar, suffer from the 'vanishing gradient' problem. Essentially, for large inputs, the resultant weight update is small (due to the bounded outputs) culminating in a slow learning rate. ReLU remedies this by having an unbounded upper limit (i.e. $[0, inf)$) and therefore is increasingly popular as a hidden layer activation function [89]. Instead, the ReLU suffers from the 'dying ReLU' problem whereby large negative weights induce a zero gradient. This means that neuron can no longer contribute to learning in the network.

Regarding weight initialization, assuming that the activation functions are the same between the neurons in a layer, the initial weights should be distinct from each other. This is done to eliminate redundancy and facilitate learning of different features [91]. Typically, the weights are initialized using a uniform distribution.

The hyperparameters which relate to the structure of the ANN, such the as number of hidden layers and the number of neurons per layer, dictate the complexity of the ANN. As discussed before, this complexity influences the approximation capabilities of the ANN. To reiterate, adding more neurons in a layer, up to a certain point, will typically reduce the error between the training data and model predictions. Adding more layers improves the abstraction capabilities of the model empowering it to process more complex input-output mappings. However, higher ANN structural complexity requires more computational resources to train and runs the risk of over-fitting the training data.

Related to the training of the ANN are the choices of the cost function, optimizer, number of training epochs and the training batch size (if any). The cost function should be chosen such that it reflects the desired convergence characteristics and that the resultant error gradient is informative for weight updates. For example, for regression, an appropriate cost function may be one that minimizes the squared error. Whereas for classification, such a cost function is not as informative as one which details the number of correct classifications. The optimizer governs the speed and convergence of the training and is therefore a trade-off between training time and convergence to (local) optima. The batch size indicates how much of the training data set should be used for training at a time while the number of epochs denotes one forward and backward pass of the entire training dataset. Therefore, the more epochs, the more the network learns and the more time it takes to train. After a certain number of epochs, the learning effectively plateaus. This can be an indicator of over-fitting since the cost function evaluates the error with respect to the training data.

Regularization

Recall that the goal of the training is to minimize the error between the model predictions and the underlying dynamics of the process in question. The cost function used in backpropagation (see section 7.2.1) actually measures the empirical loss of the model, which is the error to the *training* data. Hence, a good empirical loss does not necessarily reflect good overall model performance, since the ANN may now be capturing noise and artefacts in the training data. This is mitigated through regularization (see definition 7.2.1) techniques [91].

Definition 7.2.1

Regularization involves any technique used to improve the performance of an artificial neural network for states outside of the training data set.

Perhaps the most effective form of regularization is to simply use more real data [91]. The idea here is that, with more data, the general trends underlying the modelled process emerge over the noise and artefacts contained in the data samples. However, simply obtaining more data is often expensive and impractical. An alternative, then, is to augment the existing data to produce additional data samples [88, 91]. The principle of data augmentation is to modify the data in such a way that reflects the true data and encourages the ANN to generalize. For instance, for image classification, existing images could be modified in color, translated, or rotated to create 'unseen' images [88]. Of course, such augmentation is straightforward for classification based tasks [91] and much more difficult for regression, as is the case for system identification. Moreover, designers must be cautious when augmenting data such as to not introduce artefacts which may subsequently be modelled by the ANN. Referring back to the image recognition example, this would entail avoiding rotations of certain letters, such as 'm', to avoid confusion with others, such as 'w'.

It is typically desirable to have similar weight magnitudes across the ANN, such that few neurons do not dominate the output of the network leading to instability in these outputs. Both L1 and L2 regularization are often used to penalize the weights of an ANN [91]. These weight penalties are appended to the cost function, J, used to update the weights in the backpropagation technique. In general, this can be described by eq. 7.61 where $p = 1$ and $p = 2$ denote L1 and L2 regularization respectively, and $\lambda \in [0, \infty)$ scales the influence of the weight penalty term [91].

$$J(\mathbf{y}, \mathbf{t}, \mathbf{w}) = \bar{J}(\mathbf{y}, \mathbf{t}) + \lambda ||\mathbf{w}||_p^p \tag{7.61}$$

In L1 regularization, the weight penalties act linearly on the cost function leading to zero-weights. This results in a spare network where only the essential weights are remain, representing the essential features of the network [91]. The weight penalties in L2 regularization are quadratic with respect to the cost function, leading to small valued weights. This has implications on the variance of the input data, wherein features that have low covariance with the outputs have 'reduced' weights [91].

Regularization can also be accomplished by exploiting the stochastic training process. Ordinarily, ANNs are initialized with random weights. Consequently, the emergent model is typically different from one initialization to the next. Multiple ANNs can be trained on the same data set since the errors present in one model are unlikely to appear in another [91]. This model ensemble method has seen success in ANN literature (see, for instance, the improved ensemble performance of Krizhevsky et al. [88]). This approach is well-founded in the field of machine learning and is used to reduce the variance of the output by combining, in some way, the model ensemble outputs. This can be acheived by simply averaging the ensemble outputs. Given that some models perform better than others, Bishop [93] proposes a weighted average to improve results. More complex combinations include stacked generalization, wherein a non-linear method (even another ANN) is used to combine the model outputs [94]. In the case of ANNs of the same structure and initial weights, it is also possible to average the weights of a model ensemble to produce one model. This approach has also seen improved predictions [95]. This may be done practically by averaging the weights of the last few epochs of the training phase. It should be noted that averaging weights of networks with the same structure but different weight initializations is ill-advised and typically does not lead to improved performance.

Training a single model can be time consuming and resource intensive, thus training model ensembles may not be practical depending on the application. A more practical implementation of this concept is to use 'dropout' while training [88]. For every forward pass of the backpropagation algorithm, neurons are randomly 'dropped out' by setting their weight to zero with a certain probability. This alters the network architecture for each pass, therefore acting as a 'different' model. Since neurons can dropout, the co-dependence of neurons is reduced and they are subsequently encouraged to learn more general features [88]. Once the training is complete, all neurons are included in the model, scaled by the probability of dropping out [88].

Validation

A considerable concern for modelling with ANNs, and black-box models in general, is the issue of over-fitting the training dataset, even if the aforementioned regularization techniques are applied. Therefore, it is customary to partition the dataset into a training subset and a validation (or test) subset. The training subset is used solely for the training of the ANN while the validation set is used to evaluate the performance of the ANN using 'unseen' data. The goal, then, is to determine the level of ANN complexity (i.e. the selection of hyperparameters) which results in the lowest validation (or total) error (refer to fig. 7.4). While various error functions may be used, the mean squared error (eq. 7.62) is a popular choice and is compatible with regression.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(t[i] - \hat{y}[i])^2 \tag{7.62}$$

One may argue that the validation error could be coincidental and not reflective of the true model performance. Cross-validation is a form of validation which may be used to mitigate this. The idea is similar to that of model ensembles in that the dataset is randomly partitioned into a training and validation subset multiple times. The combination (e.g. average) of the validation errors gives a better approximation of the true model performance, for a given set of hyperparameters. When cross-validation is used, the training data subset is typically partitioned

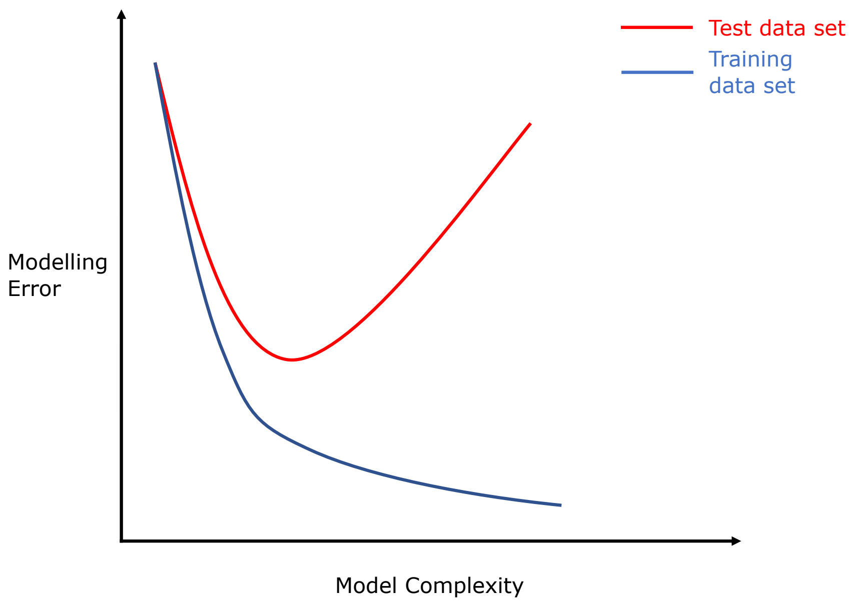


Figure 7.4: Illustration of the neural network (NN) performance, as a function of the NN complexity, with respect to the training data set (blue) and the test data set (red).

further to facilitate cross-validation. The idea is to preserve the test subset such that the model can be evaluated with respect to unseen inputs.

7.2.3 Neural networks as applied to quadrotor system identification

Through the discussion on how ANNs work and their alluring modelling capabilities, it is no surprise that ANNs have been applied for quadrotor system identification. Bansal et al. [35] successfully demonstrated, through real flight tests, the use of an ANN to facilitate the system identification of a quadrotor. Perhaps most notable from the results of [35] is the found generalization capabilities of even simple feed-forward ANNs. The quadrotor was able to track trajectories which involved simultaneous translation and (yaw) rotation when the ANN was only trained on data from the decoupled dynamics (i.e. only rotation and only translation) [35]. Therefore, to illustrate how ANNs may be employed for system identification, the approach of [35] is used as an example here.

Let the quadrotor state be defined as $\mathbf{x} = [\boldsymbol{\xi}, \mathbf{V}, \boldsymbol{\zeta}, \boldsymbol{\Omega}]^T$ where $\boldsymbol{\xi} = [x, y, z]^T$ denotes the position, $\mathbf{V} = \dot{\boldsymbol{\xi}} = [u, v, w]^T$ the velocity, $\boldsymbol{\zeta} = [\phi, \theta, \psi]^T$ the euler angles, and $\boldsymbol{\Omega} = [p, q, r]^T$ the angular velocities of the quadrotor. Contrary to the inputs defined in section 4.2, Bansal et al. [35] define the inputs to be $\mathbf{u} = [u_1, u_2, u_3, u_4]^T$ where u_1 gives the thrust, u_2 the rolling moment, u_3 the pitching moment and u_4 the yawing moment. This is synonymous with the control forces and moments definitions in section 4.2 and is therefore a linear combination of the rotor speeds (see eq. 4.12 and eq. 4.13).

The state-derivative is given by eq. 7.63 wherein $f(\mathbf{x}, \mathbf{u}; \theta)$ represents the system model, with parameters θ, mapping the states and inputs to the state-derivative. A distinction is made between the mapping from states and inputs to the derivative of velocity, denoted by f_v, and the mapping to the derivative of the angular rates, f_Ω.

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}; \theta) = \begin{bmatrix} V \\ f_v(\mathbf{x}, \mathbf{u}; \theta_1) \\ R_{EB}\boldsymbol{\Omega} \\ f_\Omega(\mathbf{x}, \mathbf{u}; \theta_2) \end{bmatrix} \tag{7.63}$$

As with any system identification task, the goal is to derive θ which minimizes the cost function for which Bansal et al. [35] elect to use the mean squared prediction error (MSE). Equation 7.64 and eq. 7.65 express the MSE for f_v and f_Ω respectively. Here, N denotes the number of observations and $\bar{f}_{(\cdot),n}$ the respective observed values (i.e. targets).

$$J_v = \underset{\theta_1}{min}\,\frac{1}{N}\left(\sum_{n=1}^{N}||\bar{f}_{v,n} - f_v(\mathbf{x}_n, \mathbf{u}_n; \theta_1)||^2\right) \tag{7.64}$$

$$J_\Omega = \underset{\theta_2}{min}\,\frac{1}{N}\left(\sum_{n=1}^{N}||\bar{f}_{\Omega,n} - f_\Omega(\mathbf{x}_n, \mathbf{u}_n; \theta_2)||^2\right) \tag{7.65}$$

Bansal et al. [35] subsequently train two multilayered feed-forward ANNs (one for each cost function). This simple network architecture is composed of an input layer which passes $\mathbf{x}$ and $\mathbf{u}$ to a single hidden layer. ReLU is chosen as the hidden layer activation function motivated by previous successes in system identification literature [35]. The output layer receives inputs from the hidden layer to which it adds a bias term to produce the ANN output, f_v (or equivalently, f_Ω). Therefore, the equation representing the output of NN_v (i.e. ANN associated with f_v) can be written succinctly as

$$f_v(\beta; \theta_1) := W_{o,v}^T \sigma(W_{h,v}^T \beta + b_{h,v}) + b_{o,v} \tag{7.66}$$

Likewise, the expression for the output of NN_Ω (i.e. ANN of f_Ω) is

$$f_\Omega(\beta; \theta_2) := W_{o,\Omega}^T \sigma(W_{h,\Omega}^T \beta + b_{h,\Omega}) + b_{o,\Omega} \tag{7.67}$$

For eq. 7.66 and eq. 7.67, $\beta := (\mathbf{x}, \mathbf{u})$ denotes the input to the network and $\sigma(\cdot)$ denotes the hidden layer ReLU activation function (i.e. $\sigma(\cdot) = \max(0, \cdot)$). W_h and W_o are the weights of the hidden and output layers respectively. Likewise, b_h and b_o are the bias vectors for the hidden and output layers respectively. These networks are trained separately using supervised learning where information on the *current* state and input is used to predict the subsequent state. Only the current state is used since this eases the design of the controller [35]. It is noteworthy that other literature sources employing ANNs for system identification, such as [83], retain knowledge of previous states in their ANN to improve model performance.

Bansal et al. [35] collect training data through automated and manual flight tests of their quadrotor resulting in $N = 240,000$ data samples. Prior to training, these data samples are preprocessed to expedite the training procedure. The trigonometric identities of the euler angles are used such that the ANNs only infer information from the orientations themselves instead of the absolute values of the angles (e.g. 0 and 2π are the same orientation). Positional inputs are not fed into the ANNs since the accelerations of the drone should be independent of these. The remaining inputs are normalized and centered around zero for better training performance. Therefore, the inputs to NN_v and NN_Ω are given by

$$\beta = [\mathbf{V}, \boldsymbol{\Omega}, \sin(\boldsymbol{\zeta}), \cos(\boldsymbol{\zeta}), u_1, u_2, u_3, u_4] \tag{7.68}$$

In the initial training phase, Bansal et al. [35] discovered that including the angular moment terms (u_2, u_3, u_4) in β for NN_v resulted in poor performance due to over-fitting. Similar observations were made with including u_1 for NN_ω. Since u_1 represents the total thrust, it is not expected, based on the first principle models, to influence the orientation of the quadrotor. Likewise, the angular moments (u_2, u_3, u_4) should not influence the translation of the quadrotor, since this should be independent of orientation. Therefore, Bansal et al. [35] remove these inputs from the respective ANNs. In reality, however, there are couplings between the orientations and translations. For instance, since the thrust direction is fixed to the body of the quadrotor, the movement of the quadrotor depends on its orientation (e.g. to move forwards the quadrotor must pitch down; it cannot move forwards while pitching up). This dependence is only exaggerated at higher velocities. Moreover, yawing while translating may lead some rotors to reach their saturation limits more quickly than expected. Perhaps the simplicity of the ANN structure proposed by [35] is unable to capture these. Incorporating more layers to the ANN may ameliorate the over-fitting issue by enabling the ANN to capture more complex interaction effects. However, these considerations were not discussed by Bansal et al. [35].

To evaluate the resultant ANNs, Bansal et al. [35] partitioned the data into three segments. The first two are used for training (60%) and (cross-)validation (25%). This promotes the generalization of the ANNs and facilitates the tuning of the hyperparameters. The remaining 15% of the collected data is used to evaluate the resultant ANNs and check for over-fitting. Since the training MSE and test MSE were similar for both NN_v and NN_Ω, it was concluded that the resultant ANNs were able to learn the dynamics to a satisfactory accuracy [35]. These ANNs were then exposed to a reference trajectory which coupled their learned dynamics (i.e. involved simultaneous rotations and translation). The results of Bansal et al. [35] show that the ANNs facilitate the control of the quadrotor for this unseen input, demonstrating their generalization capabilities with even simple network architectures.

Indeed, there are numerous strategies to implement an ANN based architecture for system identification, varying in intricacy. For instance, many researchers prefer the use of radial basis function neural networks (RBFNN) due their local properties [82]. Extensions to the RBFNN are typically focused on improving the training process [82]. While many researchers opt for simple feed-forward ANNs with one [35, 82] or two [34] hidden layers, different network architectures, such as RNNs [83], are also used. The complexity of these networks grows as research in the field matures and the computational capacity of the quadrotors improves. Moreover, ANNs may also be used in hybrid approaches. The defining characteristics of a system may be captured by simpler methods, or even through analytical approaches. The unmodelled aspects of the dynamics are then accounted for by an ANN. In fact, Mohajerin et al. [83] employed such a hybrid approach to system identification. This hybrid model consisted of the standard quadrotor model whose outputs were augmented by an RNN to account for deviations. Simulation results indicate that this hybrid model outperforms the standard model and a standalone black-box RNN model [83]. Clearly, such hybrid approaches show potential and may facilitate the development of more transparent and accurate models. The next step is to demonstrate such (hybrid) DNNs in flight tests.

8 State estimation

In order to apply system identification to the quadrotor, it is necessary to obtain information about its state. This process is facilitated by state estimation and there are a number of techniques which may be used to achieve this [16]. While not the central focus of this paper or the subsequent research, it is important to understand the capabilities and limitations of current state estimation routines. Therefore, a brief summary of commonly used sensors and state estimation algorithms are presented in this chapter.

Much of the current state-of-the-art quadrotor research, which conduct flight tests, often make use of an external motion capturing system, e.g. OptiTrack or Vicon, to assist with the state estimation (see, for instance, [1, 2, 5, 7, 19, 22]). Such systems are unrealistic for use in outdoor applications. Therefore, recent literature has focused on developing state estimation algorithms reliant only on the available on-board sensors [16, 17, 38, 96, 97].

Subsequently, section 8.1 summarizes some commonly used quadrotor sensors while section 8.2 reveals how these sensors are used in both traditional and novel state estimation algorithms.

8.1 Quadrotor sensors

Most quadrotors are commonly outfitted with an inertial measurement unit (IMU) and a camera [17, 97]. The IMU is typically composed of an accelerometer, a rate gyroscope and, occasionally, a magnetometer[†]. One of the main challenges with quadrotor state estimation is obtaining an accurate estimate of the quadrotor velocity. Quadrotor camera(s) are increasingly being used to facilitate this velocity estimate [97]. Consequently, the IMU sensors and camera will be briefly summarized in this section.

8.1.1 Accelerometer

The accelerometer is assumed to be attached near the center-of-mass of the quadrotor, and subsequently measures the specific force in the body reference frame, $\{B\}$ [16, 98]. In theory, the specific force measured by an accelerometer is given by eq. 8.1 where m denotes the mass, R_{EB} the rotation matrix from the body to the inertial frame, $\mathbf{F}$ represents the force acting on the quadrotor while $\mathbf{g}$ denotes the gravity vector.

$$\dot{\mathbf{V}} = \frac{1}{m} R_{EB} \left(\mathbf{F} - m\mathbf{g} \right) \tag{8.1}$$

In reality, the low-cost MEMS accelerometer also measures random noise, $\boldsymbol{\epsilon}_a$, and (time-varying) bias, $\mathbf{b}_{a,t}$ [16, 97, 99]. Assuming zero-mean Gaussian white noise with standard deviation $\Sigma_a = \text{diag}(\sigma_{a,x}, \sigma_{a,y}, \sigma_{a,z})$ [16, 99], the actual measured acceleration is

$$\dot{\mathbf{V}}_m = \dot{\mathbf{V}} + \boldsymbol{\epsilon}_a + \mathbf{b}_{a,t} \tag{8.2}$$

8.1.2 Rate gyroscope

The angular velocity of the quadrotor, measured in the body frame, can be obtained through the rate gyroscope. As with accelerometers, the measurement quality of low-cost MEMS rate gyroscopes is often quite poor [97, 99]. Therefore, the angular velocities are often contaminated with random noise and a time-varying bias vector [16, 97]. Assuming zero-mean Gaussian white

[†]Note that in some literature, such as [16], the magnetometer is not considered a part of the IMU. Instead, the combination of the magnetometer and IMU is termed MARG (Magnetic, Angular Rate, Gravity).

noise with a standard deviation $\Sigma_g = \text{diag}(\sigma_{g,x}, \sigma_{g,y}, \sigma_{g,z})$, the gyroscope measurement can be expressed through

$$\mathbf{\Omega}_m = \mathbf{\Omega} + \boldsymbol{\epsilon}_g + \mathbf{b}_{g,t} \tag{8.3}$$

In eq. 8.3, Ω denotes the actual angular rate, $\boldsymbol{\epsilon}_g$ the zero-mean white noise vector and, $\mathbf{b}_{g,t}$ the (time-varying) bias vector.

8.1.3 Magnetometer

Magnetometers are typically used to resolve the heading angles of a quadrotor by locally measuring the earth's magnetic field [97]. This magnetic field may be treated as a constant due to available magnetic models [16]. However, the local magnetic field may be locally disturbed due to the environment (e.g. power lines) and the systems onboard the quadrotor itself [99]. Therefore, magnetometers often require some calibration before their use [97]. Assuming this is completed, the magnetic field measured by the magnetometer can be described by

$$\mathbf{B}_m = R_{BE}\mathbf{B} + \boldsymbol{\epsilon}_b + \mathbf{b}_b \tag{8.4}$$

In eq. 8.4, $\mathbf{B}_m$ represents the (calibrated) measured magnetic field whereas $\mathbf{B}$ is the local magnetic field vector expressed in the inertial frame. Hence, R_{BE} denotes the rotation matrix from the inertial to the body reference frame. $\boldsymbol{\epsilon}_b$ gives the zero-mean Gaussian white noise recorded by the sensor with standard deviation $\Sigma_b = \text{diag}(\sigma_{b,x}, \sigma_{b,y}, \sigma_{b,z})$ and $\mathbf{b}_b$ denotes the time-varying bias vector [16].

8.1.4 Camera

The quality of cameras used on quadrotors may vary considerably. Cameras which manage high sampling rates with desirable resolutions are often expensive both in terms of capital and computational cost [97]. The exact specifications of the cameras depend on their application and environment. For example, the resolution of the camera should be high enough to resolve and track features in the environment to facilitate reliable visual odometry (i.e. obtaining position and orientation from cameras). Here, high definition cameras (720p or greater) appear to be appropriate. However, such cameras typically suffer from high latencies (> 140 ms). Instead, FPV cameras may be used which harbor lower latencies (< 40 ms) but suffer from poorer resolutions (from 400 to 1000 TVL[†]).

Regardless, much research is conducted by the robotics community on visual odometry which has stimulated many improvements in the field [96, 97]. Subsequently, there are various methods which may be employed in visual odometry. The principle behind these techniques is to either track certain features (e.g. edges, colors etc.) between consecutive frames [96, 100] or to compare these frames directly (e.g. in the frequency domain through phase correlation [101]). By observing the transformation of these features (or phase shifts), the changes in position and orientation of the quadrotor may be estimated [100].

Methods reliant on feature tracking see more frequent use for quadrotor state estimation [17, 96, 100]. Following the procedure of Shen et al. [100] as an example, the orientation can be computed using eq. 8.5 where the orientation of the current image, j, is given by R_j. R_n

[†]Note the difference between resolutions for conventional digital cameras, measured in pixels, and FPV cameras, measured in Television Lines (TVL). Pixel-based resolutions count the number of pixels along the vertical of an image, while TVL-based resolutions count the number of lines along the horizontal of an image. Hence, assuming a 16:9 aspect ratio, a 1000 TVL FPV resolution is slightly worse, in terms of pure resolution, than a 720p conventional camera.

denotes the camera orientation of image n, which is used as a reference. The rotation between image j and n is given by $R_{n,j}$.

$$R_j = R_n R_{n,j} \tag{8.5}$$

Using this orientation information, the position can be computed by minimizing eq. 8.6 [100]. Here, $\boldsymbol{\xi}_j$ denotes the position of the camera (assumed to be the same as the quadrotor) when image j was taken, $\mathcal{I}$ represents the features of j where each features' position is given by $\mathbf{p}_i$. $\mathbf{u}_{ij}$ represents the unit length feature vector of the camera in the body frame.

$$\boldsymbol{\xi}_j = \underset{\boldsymbol{\xi}_j}{\operatorname{argmin}} \sum_{i \in \mathcal{I}} \left|\left| \frac{\boldsymbol{\xi}_j - \mathbf{p}_i}{||\boldsymbol{\xi}_j - \mathbf{p}_i||} \times R_j \mathbf{u}_{ij} \right|\right|^2 \tag{8.6}$$

The stereo camera setup of Shen et al. [100] enabled the quadrotor to operate at moderate velocities ($4ms^{-1}$) and moderate roll and pitch angles (20°). However, such stereo camera setups are uncommon in quadrotors and add undesirable weight. In more recent work, Loianno et al. [96] utilized a monocular camera to facilitate state estimation and were able to sustain higher velocities ($4.5ms^{-1}$), roll and pitch angles (90°), and angular rates ($800°/s$) than Shen et al. [100]. It is not specified which of the cameras (VGA or 4k) equipped on the quadrotor platform used by Loianno et al. [96] facilitated the visual odometry. Other literature, such as [17], make use of the camera to recover only a single attitude.

However, there are some clear drawbacks with using visual odometry to measure the state. In order to improve the reliability and accuracy of these state estimates, images need to be corrected for the lens distortion and have any outliers removed. Subsequently, these computer vision algorithms are computationally expensive to run at a sufficient update rate and are susceptible to inaccuracies [17]. Moreover, the frame rate of the camera caps the (rotational) velocities that the quadrotor may achieve while maintaining reliable state estimates. This becomes problematic in rotor failure scenarios where the quadrotor is consistently spinning. Indeed, a faster camera may be employed, but increasing the number of frames demands more computational resources. Thus, ongoing research in visual odometry is aimed at reducing this computational cost [17]. Moreover, visual odometry in quadrotor applications relies heavily on tracking (static) features in the environment and is therefore less reliable on relatively feature-less applications (e.g. search and rescue at sea) or those where perceived motion in the environment is independent of the quadrotor's motion (e.g. moving animals) [97].

8.2 State estimation techniques

Clearly, the measurements obtained from the primary quadrotor sensors described in section 8.1 are contaminated with noise and biases. Using the states directly from these measurements may lead to ill-informed control inputs. Therefore, state estimation techniques may be employed to process, filter, and combine the measurements from these sensors to improve these state estimates. The Mahony, Madgwick and Extended Kalman Filters are commonly used for the attitude estimation of a quadrotor [99]. Consequently, these filters are summarized in this section. However, given the limitations of these filters, they underperform when tasked with estimating the state of a quadrotor with a failed motor [16]. Therefore, a novel state estimation technique proposed by Solanki [16] - designed to accommodate state estimation under such failures - is also presented.

8.2.1 Mahony filter

To enable to attitude estimation using low-cost MEMS technology, Mahony et al. [102] propose the use of three nonlinear complementary filters, collectively known as the Mahony filter. This filter utilizes two vectors defined in the earth reference frame to construct an instantaneous measurement of the rotation matrix between the inertial and body reference frames [102]. These vectors are typically taken from the accelerometer (gravity vector) and magnetometer (earth magnetic field vector) with the requirement that these vectors are non-colinear [102]. It is further assumed that the gravity vector is dominant in the acceleration measurements, which is not necessarily the case in the operational regime of the quadrotor [99] or following damages to the system [16].

However, when that these assumptions hold (e.g. during hover), the attitude estimate is obtained by integrating the rate gyroscope measurements. This alone provides a poor estimate of the attitude due to the signal quality of the rate gyroscope. Therefore, the accelerometer and magnetometer readings are used to derive the orientation error to correct the gyroscope estimates [16, 102]. This culminates in accurate orientation estimates [99], which is desirable for quadrotor applications. It is noteworthy to highlight that changes in the local magnetic field may induce errors in these estimates.

In subsequent work, Mahony et al. [103] demonstrated that the filter may still be employed when only one vector is known, provided that the direction of this vector varies over time. Moreover, much of the success of the Mahony filter can be attributed to its computational efficiency, in comparison to more conventional complementary filters [16, 99, 104].

8.2.2 Madgwick filter

Similar to the Mahony filter, the Madgwick filter also aims to tackle the challenge of attitude estimation through low-cost MEMS in a computationally efficient manner. To avoid singularities and approximation issues associated with Euler angles, Madgwick et al. [104] opt for a quaternion representation of attitude.

The principle behind the Madgwick filter is to fuse the orientation estimate as obtained through the numerical integration of the angular rates, q_{Ω}, with that obtained from aligning a reference frame with the measurements through gradient descent, q_{∇}. In doing so, the characteristic drift affecting the gyroscope measurements and the high-frequency errors stemming from the gradient descent are symbiotically accounted for [104]. The angular rates may simply be obtained from the gyroscope and subsequently integrated for q_{Ω}. The other estimate of orientation is obtained by first defining reference directions in the inertial frame, given by gravity vector and the earth's magnetic field vector, and comparing these to the measured accelerations and magnetic fields from the sensors. This results in an optimization problem wherein the objective is to find a rotation vector, q_{∇}, which aligns the reference directions with those measured [104].

However, the issue with defining a reference direction using the earth's magnetic field is that there are distortions present. Local disturbances (e.g. due to nearby metal structures) further affect the local magnetic field. Static local sources can typically be removed through calibration while distortions in the earth's magnetic field can be partially compensated for by using gravity as a reference of orientation [104]. Doing so prompts the magnetic field disturbances to only affect the yaw estimates of the attitude. This is different than the Mahony filter where disturbances affect the entire attitude estimation.

The two attitude estimates (i.e. q_{Ω} and q_{∇}) are combined by first removing the gyroscope measurement error from the rate of change of q_{Ω} (i.e $\dot{q}_{\Omega}$) along the direction defined by the magnetometer and accelerometer to obtain $\dot{q}_{est}$. The attitude estimate is then updated based on the estimate at the previous time sample and the propagation of the rate of change of orientation (i.e. $\dot{q}_{est}$) within this time interval [104].

As with the Mahony filter, the Madgwick filter also assumes that the gravity vector is the dominant acceleration in the system and that the magnetic field is constant [16, 104]. Therefore, the performance of these filters may deteriorate under conditions of prolonged body accelerations and changing magnetic fields.

8.2.3 Kalman filter

The Kalman filter is widely used in state estimation, especially for autonomous systems [105, 98, 104]. The principle of this approach is to take a weighted average of the predicted and measured state (see eq. 8.7). Specifically, the predicted state is corrected by the measurements. Since predicted states are used, some knowledge of the system model is necessary.

$$\hat{\mathbf{x}} = \mathbf{x}_p + K\left(\mathbf{z}_m - \mathbf{z}_p\right) \tag{8.7}$$

In eq. 8.7, $\hat{\mathbf{x}}$ and $\mathbf{x}_p$ denote the state estimate and state prediction respectively. Similarly, $\mathbf{z}_m$ and $\mathbf{z}_p$ represent the measured and predicted outputs of the system. The importance of these outputs is scaled by the Kalman gain, K. The measurements themselves may be improved by fusing information from multiple sensors (e.g. camera) at the cost of consuming more computational resources [106].

One of the main attractions of the Kalman filter is that it provides insights into the confidence of the state predictions and estimations through the state prediction error covariance matrix and state estimation error covariance matrix respectively. Smaller covariances reflect more reliable state predictions and/or estimates. This is assuming that the process noise and measurement noise are independent and can be represented as zero-mean Gaussian white noise [105]. Therefore, knowledge of the noise acting on the system is also required to apply the Kalman filter.

The original (linear) kalman filter is only valid for linear systems [16, 105, 107]. This is typically not the case for the quadrotor, especially for high-speed conditions. A more suitable alternative is the Extended Kalman Filter (EKF) or the Unscented KF (UKF) for especially non-linear systems. The EKF linearizes the system around the nominal values of each measurement point through the derivation of the Jacobians of the system which may become tedious for highly non-linear systems [16, 107, 108]. Subsequently, they may prove challenging to implement for some systems and can be especially computationally restrictive for MEMS-enabled systems [104]. Moreover, the error covariance matrices computed by the EKF likely underestimate the true errors since the approximation errors stemming from the linearization are neglected. Regardless, the EKF (and its variants) remain a popular and proven choice for state estimation for quadrotors.

8.2.4 Adaptive fuzzy complementary kalman filter

Solanki [16] recently developed a novel attitude estimation routine - called the Adaptive Fuzzy Complementary Kalman Filter (AFCKF) - aimed at providing accurate state estimates of the quadrotor, even under circumstances of actuator failure. As with the Mahony and Madgwick filters, the AFCKF relies only on measurements from the magnetometer, accelerometer and rate gyroscope. Despite the additional computational demand over the Madgwick filter, the AFCKF is more computationally efficient than the EKF [16].

Inspired by the Magdwick and Mahony filters, the AFCKF also relies on the gravity vector for attitude estimation. Hence, linear accelerations are assumed to be negligible in comparison to the gravity vector. However, unlike the Madgwick or Mahony filters, the AFCKF also accounts for any centrifugal accelerations measured by the accelerometer due to the offset of this sensor and the body's center of mass [16]. This correction term is necessary since, under single rotor failure conditions, the quadrotor spins about its yaw axis [21, 22] culminating in a measured centrifugal force. The acceleration can then be transformed into an attitude estimate for roll and pitch using gradient descent. This process is similar to that of the Madgwick filter [104]. Note that the yaw estimate is unreliable at this point, since the magnetometer is not (yet) used for attitude estimation. Additional attitude estimates for the roll and pitch are obtained through the numerical integration of the rate gyroscope measurements and is therefore reminiscent of Madgwick et al. [104].

Subsequently, a Multiplicative Extended Kalman Filter (MEKF) is used to fuse these two attitude estimates. The standard EKF is not compatible with a quaternion based representation of attitude without some modification [109]. To this end, Solanki [16] makes use of a Multiplicative EKF (MEKF) wherein the correction term is composed of a quaternion multiplication, which preserves their norm [109]. The MEKF is comprised of a sensor and process model whereby the sensor model relates to the attitude estimates obtained from the accelerometer and the process model those from the rate gyroscopes [16].

Instead of implementing the gyroscope bias and drift corrections in the MEKF, Solanki [16] instead opts to correct for this using an approach similar to Mahony et al. [102]. While the MEKF can accommodate these corrections, doing so increases the number of states to be estimated and thus the computational demand. The proposed correction utilizes information from the accelerometer and previous attitude estimate to account for the drift [16].

Contrary to the approaches of Mahony et al. [102] and Madgwick et al. [104] where the magnetometer is used for the full attitude estimate, Solanki [16] only uses the magnetometer and rate gyroscope to obtain the yaw estimate. The presumed benefit of such sensor separation is that any magnetic disturbances do not affect the roll and pitch attitudes, which are more crucial for control. However, the benefits of sensor fusion with the magnetometer are indeed lost for the roll and pitch estimates. The final yaw estimate is obtained through a weighted average of the yaw estimate derived from the gyroscope with that from the magnetometer. The weights of this average are modulated through fuzzy logic and vary depending on the gyroscope saturation and magnetic field disturbances [16]. The general idea being that the weight of the respective yaw estimate decreases with increasing gyroscope saturation or increasing magnetic field disturbances. Indeed, if both sensors are deemed unreliable (i.e. saturated gyroscope *and* large magnetic disturbances), then the yaw estimate is also unreliable.

9 Research proposal

Over the past decade, a considerable amount of research has been conducted on the increasingly autonomous quadrotor. Such research has demonstrated the capabilities and versatility of this vehicle by expanding upon its flight envelope. Much literature (e.g. [33, 36, 39]) builds upon the simple quadrotor model in an analytical manner by incorporating well-documented aerodynamic effects. When faced with the challenge of modelling unfamiliar phenomena, some literature (e.g. [6, 7, 35, 83]) consult data-driven system identification techniques, such as Neural Networks, to capture these effects. Others instead prefer to employ robust controllers to accommodate modelling errors and uncertainties (e.g. [4, 40, 43]). Several advancements have also been made in improving the safety of quadrotors. Most notably, the sustained control of a conventional quadrotor despite the complete loss of one and two (opposing) rotors has been demonstrated through flight tests in both low-speed [21, 22] and high-speed conditions [1, 2].

Despite these incredible results, much is still unknown about the dynamics behind the quadrotor, especially in the high-speed regime. Current quadrotors are able to operate at such conditions by virtue of the capabilities of the employed controllers. The performance of these controllers can be improved through more accurate models, thereby extending the capacity of the quadrotor. Therefore, current gaps in literature are summarized in section 9.1 with the subsequent research objectives aimed at addressing some of these gaps given in section 9.2.

9.1 Summary of gaps in literature

It is evident that quadrotors are a popular outlet for research in the fields of robotics and aerospace. While several advancements have been made over recent decades, there are still some substantial gaps in literature. Perhaps the most prominent is that much of the current research is conducted indoors, in conditions devoid of external (stochastic) disturbances, and where external motion capturing systems are usable.

Many simplifications and assumptions are made in the analytical models of the simple quadrotor and extensions thereof (i.e. chapter 4 and chapter 5 respectively). This means that the simple quadrotor model is only valid in near-hover conditions. While the extensions seek to remedy this, they themselves impose some restrictions on their use. Much of the theory behind the effects of thrust variance and blade flapping is founded in helicopter literature [36]. As such, the thrust variance effect considers the rotors individually and therefore neglects the interaction effects between these rotors. As observed in the wind tunnel flight tests of Sun et al. [7], there is indeed an interaction effect between the rotors in high-speed conditions influencing the total thrust produced. Likewise, Powers et al. [33] witnessed a divergence between their model predictions and true flight tests, likely due to similar effects. Many of the existing models in literature have only been validated in the low-speed domain of the flight envelope and are therefore unsuitable for high-speed scenarios or those involving aggressive manoeuvres. Moreover, current knowledge of these aerodynamic effects is asymmetrical in that the aerodynamic forces are well-documented whereas knowledge on aerodynamic moments is lacking. *Therefore, valid models in the high-speed regime which consider aerodynamic effects and interactions are necessary.*

Indeed, existing literature has managed to identify high-speed quadrotor models [1, 6, 7] or has successfully demonstrated aggressive manoeuvres [9, 10, 96]. System identification techniques (chapter 7) are typically used construct high-speed models while aggressive manoeuvres are facilitated by specialized controllers specific to certain trajectories or through learning. However, data collected for these endeavours are often obtained indoors, with the aid of external motion capturing systems, and are devoid of the external disturbances commonly seen in outdoor environments. Consequently, identification procedures valid for outdoor data must be resilient against both measurement and system noise. Moreover, the physical constraints of the indoor space limit the achievable speeds and manoeuvres of the quadrotor thereby restricting flight envelope coverage. *Hence, there is a need for a quadrotor model that is identifiable without the help of an external motion capturing system and is valid outdoors, in high-speed conditions, and for aggressive manoeuvres.*

Recent literature has been especially concerned with fault tolerant control (chapter 6) of quadrotors, since safety is essential for their accepted use outdoors. In fact, Stephan et al. [68] demonstrated the flight of a quadrotor with one failed rotor outdoors in low-speed conditions. Regarding high-speed flight tests, Sun et al. managed to achieve high-speed flight of a quadrotor with one failed rotor [1] and, more recently, two opposing failed rotors [2] by maintaining the so-called relaxed trimming equilibrium. However, this equilibrium is broken when the quadrotor experiences sustained accelerations, such as during aggressive manoeuvres. In this literature [1, 2, 68], failed rotors were imposed before the quadrotor began motion and, in the case of Sun et al. [1, 2], were removed entirely. Moreover, while theoretical solutions to adjacent double rotor failures exist in literature, none have managed to validate these in flight. This failure mode is arguably more common than the diagonal rotor failure scenario investigated by Sun et al. [2]. *Thus, it is currently unclear how failures incurred during high-speed flight, and how the presence of the failed rotor(s), influence the quadrotor - especially for adjacent double rotor failures.*

However, before this can be determined, *the effect of a partial loss of effectiveness of one, or multiple, rotors sustained during high-speed flight should be investigated.* In literature, partial faults are accommodated either through modelling [18, 41, 44] or a fault tolerant controller [4, 43, 46]. While utilizing fault tolerant controllers is more popular, many of the employed controllers result in reduced performance and are sensitive to external disturbances. Likewise, approaches related to modelling the faults make use of fault estimation and detection algorithms which also suffer from similar sensitives [64]. This makes them ill-suited for applications outdoors where external disturbances are prevalent. *Therefore, an adaptive framework composed of online model identification and fault accommodation is necessary to address partial failures, and other non-stationary effects, influencing the quadrotor in outdoor flight.*

9.2 Research objectives

In many ways, the potential of a quadrotor is constrained by its model and controller. While much research is conducted on improving these controllers, less attention is directed towards the creation of better models, even though these can also facilitate substantial advancements in performance. Many applications of quadrotors see their routine operation in outdoor environments, often at high-speeds or demanding aggressive manoeuvres. In existing literature, high fidelity quadrotor models valid under such conditions have yet to be identified using information solely from the on-board sensors.

Accordingly, the proposed research objective relates to the system identification of quadrotors. Since a high fidelity outdoor model of the quadrotor has never been identified due to the challenges it entails, the proposed research objective limits itself to the nominal quadrotor and does not enforce an online implementation. Nevertheless, in consideration of these ultimate goals, this research objective is also concerned with the efficiency of potential system identification routines. Therefore, the proposed research objective is

Towards an online system identification routine for an undamaged quadrotor suitable for high-speed outdoor applications and aggressive manoeuvres using only on-board sensor information.

In light of recent literature on system identification (e.g. [7, 83]), a hybrid approach between stepwise regression and artificial neural networks (ANN) shows promise for outdoor system identification and will be an integral aspect of the subsequent research. The concept behind the proposed approach is for the stepwise regression to capture the dominant stationary (body) dynamics while the ANN compensates for more complex, perhaps non-stationary, effects. Therefore, in order to tackle the stated research objective, the following research questions should be addressed:

i. How can (combinations of) commercially available quadrotor sensors be used for system identification in visual tracking system denied environments?
    (a) What are the necessary on-board sensors to facilitate state estimation?
    (b) How may these sensors be configured to reliably and repeatedly measure the relevant quadrotor states?

ii. How does the proposed hybrid system identification approach compare to its constituent system identification techniques individually?

iii. What type of flight manoeuvres are necessary to excite the quadrotor body dynamics?
    (a) What manoeuvres are measurable in outdoor environments by the equipped sensors?
    (b) How can these manoeuvres be reliably and repeatedly performed in outdoor environments?

iv. How feasible is the proposed system identification routine for an online implementation?
    (a) What is the computational complexity of the proposed hybrid system identification algorithm?
    (b) What requirements would the proposed hybrid system identification approach impose on the quadrotor hardware?

References

[1] S. Sun, L. Sijbers, X. Wang, and C. de Visser. “High-Speed Flight of Quadrotor Despite Loss of Single Rotor”. In: *IEEE Robotics and Automation Letters* 3.4 (Oct. 2018), pp. 3201–3207. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2851028.

[2] S. Sun, X. Wang, Q. Chu, and C. d. Visser. “Incremental Nonlinear Fault-Tolerant Control of a Quadrotor With Complete Loss of Two Opposing Rotors”. In: *IEEE Transactions on Robotics* (2020), pp. 1–15. ISSN: 1941-0468. DOI: 10.1109/TRO.2020.3010626.

[3] S. Sun and C. de Visser. “Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime”. In: *IEEE Robotics and Automation Letters* 4.4 (Oct. 2019), pp. 3868–3875. ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2928758.

[4] X. Wang, S. Sun, E.-J. van Kampen, and Q.P. Chu. “Quadrotor Fault Tolerant Incremental Sliding Mode Control driven by Sliding Mode Disturbance Observers”. In: *Aerospace Science and Technology* 87 (2019), pp. 417–430. ISSN: 1270-9638. DOI: 10.1016/j.ast.2019.03.001.

[5] S. Sun, M. Baert, B. S. van Schijndel, and C. C. de Visser. “Upset Recovery Control for Quadrotors Subjected to a Complete Rotor Failure from Large Initial Disturbances”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. May 2020, pp. 4273–4279. DOI: 10.1109/ICRA40945.2020.9197239.

[6] S. Sun, R. J. Schilder, and C. C. de Visser. “Identification of Quadrotor Aerodynamic Model from High Speed Flight Data”. In: *2018 AIAA Atmospheric Flight Mechanics Conference*. 2018. DOI: 10.2514/6.2018-0523.

[7] S. Sun, C.C. de Visser, and Q.P. Chu. “Quadrotor Gray-Box Model Identification from High-Speed Flight Data”. In: *Journal of Aircraft* 56.2 (2019), pp. 645–661. DOI: 10.2514/1.C035135.

[8] S. Li, C. De Wagter, C.C. de Visser, Q.P. Chu, and G.C.H.E. de Croon. “In-flight model parameter and state estimation using gradient descent for high-speed flight”. In: *International Journal of Micro Air Vehicles* 11 (2019). DOI: 10.1177/1756829319833685.

[9] D. Mellinger, N. Michael, and V. Kumar. “Trajectory generation and control for precise aggressive maneuvers with quadrotors”. In: *The International Journal of Robotics Research* 31.5 (2012), pp. 664–674. DOI: 10.1177/0278364911434236.

[10] D. Molenkamp, E.-J. Van Kampen, C. C. de Visser, and Q. P. Chu. “Intelligent Controller Selection for Aggressive Quadrotor Manoeuvring”. In: *AIAA Information Systems-AIAA Infotech @ Aerospace*. DOI: 10.2514/6.2017-1068.

[11] E. J. J. Smeur, Q.P. Chu, and G. C. H. E. de Croon. “Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Air Vehicles”. In: *Journal of Guidance, Control, and Dynamics* 39.3 (2016), pp. 450–461. DOI: 10.2514/1.G001490.

[12] S. Sun and C.C. de Visser. “Quadrotor Safe Flight Envelope Prediction in the High-Speed Regime: A Monte-Carlo Approach”. In: *AIAA Scitech 2019 Forum*. DOI: 10.2514/6.2019-0948.

[13] J. K. Stolaroff, C. Samaras, E. R. O’Neill, A. Lubers, A. S. Mitchell, and D. Ceperley. “Energy use and life cycle greenhouse gas emissions of drones for commercial package delivery”. In: *Nature communications* 9.1 (2018), pp. 1–13. DOI: https://doi.org/10.1038/s41467-017-02411-5.

[14] J. Verbeke and J. De Schutter. “Experimental maneuverability and agility quantification for rotary unmanned aerial vehicle”. In: *International Journal of Micro Air Vehicles* 10.1 (2018), pp. 3–11. DOI: `https://doi.org/10.1177/1756829317736204`.

[15] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin. “Quadrotor helicopter flight dynamics and control: Theory and experiment”. In: *AIAA guidance, navigation and control conference and exhibit.* 2007, p. 6461. DOI: `10.2514/6.2007-6461`.

[16] P. Solanki. “Attitude Estimation of a Quadcopter with one fully damaged rotor using on-board MARG Sensors”. MA thesis. Kluyverweg 1, 2629 HS Delft, The Netherlands: Delft University of Technology, Oct. 2020. URL: `http://resolver.tudelft.nl/uuid:43b7a1c8-7b67-48a4-97b0-7fe9c8af884a`.

[17] J. Svacha, G. Loianno, and V. Kumar. “Inertial Yaw-Independent Velocity and Attitude Estimation for High-Speed Quadrotor Flight”. In: *IEEE Robotics and Automation Letters* 4.2 (Apr. 2019), pp. 1109–1116. ISSN: 2377-3766. DOI: `10.1109/LRA.2019.2894220`.

[18] R. C. Avram, X. Zhang, and J. Muse. “Quadrotor Actuator Fault Diagnosis and Accommodation Using Nonlinear Adaptive Estimators”. In: *IEEE Transactions on Control Systems Technology* 25 (6 Nov. 2017), pp. 2219–2226. ISSN: 10636536. DOI: `10.1109/TCST.2016.2640941`.

[19] R. C. Avram, X. Zhang, and J. Muse. “Nonlinear Adaptive Fault-Tolerant Quadrotor Altitude and Attitude Tracking with Multiple Actuator Faults”. In: *IEEE Transactions on Control Systems Technology* 26 (2 Mar. 2018), pp. 701–707. ISSN: 10636536. DOI: `10.1109/TCST.2017.2670522`.

[20] A.-R. Merheb, H. Noura, and F. Bateman. “Design of Passive Fault–Tolerant Controllers of a Quadrotor Based on Sliding Mode Theory”. In: *International Journal of Applied Mathematics and Computer Science* 25 (Sept. 2015). DOI: `10.1515/amcs-2015-0042`.

[21] M. W. Mueller and R. D’Andrea. “Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers”. In: *2014 IEEE international conference on robotics and automation (ICRA).* IEEE. 2014, pp. 45–52. ISBN: 978-1-4799-3685-4. DOI: `10.1109/ICRA.2014.6906588`.

[22] M. W. Mueller and R. D’Andrea. “Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles”. In: *International Journal of Robotics Research* 35 (8 2016), pp. 873–889. ISSN: 17413176. DOI: `10.1177/0278364915596233`.

[23] S. Bouabdallah, P. Murrieri, and R. Siegwart. “Design and Control of an Indoor Micro Quadrotor”. In: *Proc. of The International Conference on Robotics and Automation (ICRA).* Vol. 5. IEEE International Conference on Robotics and Automation (ICRA 2004). Zürich: ETH-Zürich, 2004, pp. 4393–4398. DOI: `10.3929/ethz-a-010085499`.

[24] J. Jiang, J. Qi, D. Song, and J. Han. “Control platform design and experiment of a quadrotor”. In: *Proceedings of the 32nd Chinese Control Conference.* July 2013, pp. 2974–2979. ISBN: 978-9-8815-6383-5.

[25] J. G. Leishman. “The breguet-richet quad-rotor helicopter of 1907”. In: *Vertiflite* 47.3 (2002), pp. 58–60.

[26] J. Kim, S. A. Gadsden, and S. A. Wilkerson. “A Comprehensive Survey of Control Strategies for Autonomous Quadrotors”. In: *Canadian Journal of Electrical and Computer Engineering* 43.1 (2020), pp. 3–16. ISSN: 0840-8688. DOI: `10.1109/CJECE.2019.2920938`.

[27] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin. “Precision flight control for a multi-vehicle quadrotor helicopter testbed”. In: *Control Engineering Practice* 19.9 (2011). Special Section: DCDS’09 – The 2nd IFAC Workshop on Dependable Control of Discrete Systems, pp. 1023–1036. ISSN: 0967-0661. DOI: https://doi.org/10.1016/j.conengprac.2011.04.005.

[28] C. Gablehouse. *Helicopters and Autogiros: A History of Rotating-wing and V/STOL Aviation.* English. Philadelphia: Lippincott, 1969.

[29] S. Harding. *U.S. Army Aircraft Since 1947: An Illustrated Reference.* Schiffer Military History. Schiffer Pub., 1997. ISBN: 9780764301902.

[30] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin. “The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC)”. In: *The 23rd Digital Avionics Systems Conference (IEEE Cat. No.04CH37576).* Vol. 2. Oct. 2004, 12.E.4–121. DOI: 10.1109/DASC.2004.1390847.

[31] B. Gati. “Open source autopilot for academic research - The Paparazzi system”. In: *2013 American Control Conference.* June 2013, pp. 1478–1481. DOI: 10.1109/ACC.2013.6580045.

[32] J. Svacha, K. Mohta, and V. Kumar. “Improving quadrotor trajectory tracking by compensating for aerodynamic effects”. In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS).* June 2017, pp. 860–866. DOI: 10.1109/ICUAS.2017.7991501.

[33] C. Powers, D. Mellinger, A. Kushleyev, B. Kothmann, and V. Kumar. “Influence of aerodynamics and proximity effects in quadrotor flight”. In: *Experimental Robotics: The 13th International Symposium on Experimental Robotics.* Heidelberg: Springer International Publishing, 2013, pp. 289–302. ISBN: 978-3-319-00065-7. DOI: 10.1007/978-3-319-00065-7_21.

[34] N. A. Bakshi and R. Ramachandran. “Indirect model reference adaptive control of quadrotor UAVs using neural networks”. In: *Proceedings of the 10th International Conference on Intelligent Systems and Control, ISCO 2016.* Institute of Electrical and Electronics Engineers Inc., Oct. 2016. ISBN: 9781467378079. DOI: 10.1109/ISCO.2016.7727123.

[35] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin. “Learning quadrotor dynamics using neural network for flight control”. In: *2016 IEEE 55th Conference on Decision and Control, CDC 2016.* Institute of Electrical and Electronics Engineers Inc., Dec. 2016, pp. 4653–4660. ISBN: 9781509018376. DOI: 10.1109/CDC.2016.7798978.

[36] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin. “Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering”. In: *2009 IEEE international conference on robotics and automation.* IEEE. 2009, pp. 3277–3282. ISBN: 978-1-4244-2788-8. DOI: 10.1109/ROBOT.2009.5152561.

[37] W. Johnson. *Helicopter Theory.* Dover Books on Aeronautical Engineering. Dover Publications, 2012. ISBN: 9780486131825. URL: https://books.google.nl/books?id=FiEapaNgjLcC.

[38] D. Abeywardena, S. Kodagoda, G. Dissanayake, and R. Munasinghe. “Improved State Estimation in Quadrotor MAVs: A Novel Drift-Free Velocity Estimator”. In: *IEEE Robotics Automation Magazine* 20.4 (2013), pp. 32–39. DOI: 10.1109/MRA.2012.2225472.

[39] R. Mahony, V. Kumar, and P. Corke. “Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor”. In: *IEEE Robotics Automation Magazine* 19.3 (Aug. 2012), pp. 20–32. ISSN: 1558-223X. DOI: 10.1109/MRA.2012.2206474.

[40] A. Chamseddine, Y. Zhang, C. A. Rabbath, C. Fulford, and J. Apkarian. “Model reference adaptive fault tolerant control of a quadrotor UAV”. In: *AIAA Infotech at Aerospace Conference and Exhibit 2011*. American Institute of Aeronautics and Astronautics Inc., 2011. ISBN: 9781600869440. DOI: 10.2514/6.2011-1606.

[41] A. Baldini, R. Felicetti, A. Freddi, A. Monteriu, and M. Tempesta. “Estimation of actuator faults in quadrotor vehicles: from theory to validation with experimental flight data”. In: *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. Institute of Electrical and Electronics Engineers (IEEE), Oct. 2020, pp. 1249–1256. ISBN: 978-1-7281-4278-4. DOI: 10.1109/icuas48674.2020.9214026.

[42] T. Li, Y. Zhang, and B. W. Gordon. “Passive and active nonlinear fault-tolerant control of a quadrotor unmanned aerial vehicle based on the sliding mode control technique”. In: *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 227.1 (2013), pp. 12–23. DOI: 10.1177/0959651812455293.

[43] S. Xiao and J. Dong. “Robust Adaptive Fault-Tolerant Tracking Control for Uncertain Linear Systems with Actuator Failures Based on the Closed-Loop Reference Model”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 50 (9 Sept. 2020), pp. 3448–3455. ISSN: 21682232. DOI: 10.1109/TSMC.2018.2876125.

[44] G. Ortiz-Torres, P. Castillo, F. D.J. Sorcia-Vázquez, J. Y. Rumbo-Morales, J. A. Brizuela-Mendoza, J. De La Cruz-Soto, and M. Martínez-García. “Fault Estimation and Fault Tolerant Control Strategies Applied to VTOL Aerial Vehicles with Soft and Aggressive Actuator Faults”. In: *IEEE Access* 8 (2020), pp. 10649–10661. ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2963693.

[45] Z. Zhao, X. Wang, P. Yao, J. Xu, and J. Yu. “Fuzzy health degree-based dynamic performance evaluation of quadrotors in the presence of actuator and sensor faults”. In: *Nonlinear Dynamics* 95 (3 Feb. 2019), pp. 2477–2490. ISSN: 1573269X. DOI: 10.1007/s11071-018-4711-2.

[46] Y.M. Zhang, A. Chamseddine, C.A. Rabbath, B.W. Gordon, C.-Y. Su, S. Rakheja, C. Fulford, J. Apkarian, and P. Gosselin. “Development of advanced FDD and FTC techniques with application to an unmanned quadrotor helicopter testbed”. In: *Journal of the Franklin Institute* 350.9 (2013), pp. 2396–2422. ISSN: 0016-0032. DOI: 10.1016/j.jfranklin.2013.01.009.

[47] L. Besnard, Y. B. Shtessel, and B. Landrum. “Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer”. In: *Journal of the Franklin Institute* 349.2 (2012). Advances in Guidance and Control of Aerospace Vehicles using Sliding Mode Control and Observation Techniques, pp. 658–684. ISSN: 0016-0032. DOI: 10.1016/j.jfranklin.2011.06.031.

[48] I. Sadeghzadeh, A. Mehta, and Y. Zhang. “Fault/damage tolerant control of a quadrotor helicopter UAV using model reference adaptive control and gain-scheduled PID”. In: *AIAA Guidance, Navigation, and Control Conference 2011*. American Institute of Aeronautics and Astronautics Inc., 2011. ISBN: 9781600869525. DOI: 10.2514/6.2011-6716.

[49] K. D. Young, V. I. Utkin, and U. Ozguner. “A control engineer’s guide to sliding mode control”. In: *IEEE Transactions on Control Systems Technology* 7.3 (May 1999), pp. 328–342. ISSN: 1558-0865. DOI: 10.1109/87.761053.

[50] Y. Shtessel, C. Edwards, L. Fridman, and A. Levant. *Sliding mode control and observation.* Springer, 2014. ISBN: 978-0-8176-4892-3.

[51] J. Ni, C. Liu, and H. Liu. "Continuous uniformly finite time exact disturbance observer based control for fixedtime stabilization of nonlinear systems with mismatched disturbances". In: *PLoS ONE* 12 (4 Apr. 2017). ISSN: 19326203. DOI: `10.1371/journal.pone.0175645`.

[52] P. Ordaz, M. Ordaz, C. Cuvas, and O. Santos. "Reduction of matched and unmatched uncertainties for a class of nonlinear perturbed systems via robust control". In: *International Journal of Robust and Nonlinear Control* 29 (8 May 2019), pp. 2510–2524. ISSN: 1049-8923. DOI: `10.1002/rnc.4506`.

[53] M. Rubagotti, A. Estrada, F. Castaños, A. Ferrara, and L. Fridman. "Integral sliding mode control for nonlinear systems with matched and unmatched perturbations". In: *IEEE Transactions on Automatic Control* 56 (11 Nov. 2011), pp. 2699–2704. ISSN: 00189286. DOI: `10.1109/TAC.2011.2159420`.

[54] V. Utkin and H. Lee. "Chattering problem in sliding mode control systems". In: *Proceedings of the 2006 International Workshop on Variable Structure Systems, VSS'06.* 2006, pp. 346–350. ISBN: 1424402085. DOI: `10.1109/VSS.2006.1644542`.

[55] P. Lu and E.-J. Van Kampen. "Active fault-tolerant control for quadrotors subjected to a complete rotor failure". In: *IEEE International Conference on Intelligent Robots and Systems* 2015-Decem (2015), pp. 4698–4703. ISSN: 21530866. DOI: `10.1109/IROS.2015.7354046`.

[56] X. Zhang, Y. Zhang, C.-Y. Su, and Y. Feng. "Fault Tolerant Control for Quadrotor via Backstepping Approach". In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition* (2010). DOI: `10.2514/6.2010-947`.

[57] P. van Gils, E.-J. Van Kampen, C. C. de Visser, and Q. P. Chu. "Adaptive Incremental Backstepping Flight Control for a High-Performance Aircraft with Uncertainties". In: *AIAA Guidance, Navigation, and Control Conference.* DOI: `10.2514/6.2016-1380`.

[58] M. Krstic, P. V. Kokotovic, and I. Kanellakopoulos. *Nonlinear and Adaptive Control Design.* 1st. USA: John Wiley & Sons, Inc., 1995. ISBN: 0471127329.

[59] L. Sonneveldt, Q. P. Chu, and J. A. Mulder. "Nonlinear flight control design using constrained adaptive backstepping". In: *Journal of Guidance, Control, and Dynamics* 30 (2 May 2007), pp. 322–336. ISSN: 15333884. DOI: `10.2514/1.25834`.

[60] B. J. Jeon, M. G. Seo, H. S. Shin, and A. Tsourdos. "Understandings of Classical and Incremental Backstepping Controllers with Model Uncertainties". In: *IEEE Transactions on Aerospace and Electronic Systems* 56 (4 Aug. 2020), pp. 2628–2641. ISSN: 15579603. DOI: `10.1109/TAES.2019.2952631`.

[61] W. Hao and B. Xian. "Nonlinear adaptive fault-tolerant control for a quadrotor UAV based on immersion and invariance methodology". In: *Nonlinear Dynamics* 90 (4 Dec. 2017), pp. 2813–2826. ISSN: 1573269X. DOI: `10.1007/s11071-017-3842-1`.

[62] X. Zhang and Y. Zhang. "Fault Tolerant Control for Quad-rotor UAV by Employing Lyapunov-based Adaptive Control Approach". In: *AIAA Guidance, Navigation, and Control Conference* (2010). DOI: `10.2514/6.2010-8052`.

[63] Y. Song, L. He, D. Zhang, J. Qian, and J. Fu. "Neuroadaptive Fault-Tolerant Control of Quadrotor UAVs: A More Affordable Solution". In: *IEEE Transactions on Neural Networks and Learning Systems* 30 (7 July 2019), pp. 1975–1983. ISSN: 21622388. DOI: `10.1109/TNNLS.2018.2876130`.

[64] X. Nian, W. Chen, X. Chu, and Z. Xu. “Robust adaptive fault estimation and fault tolerant control for quadrotor attitude systems”. In: *International Journal of Control* 93 (3 Mar. 2020), pp. 725–737. ISSN: 0020-7179. DOI: 10.1080/00207179.2018.1484573.

[65] A. Lanzon, A. Freddi, and S. Longhi. “Flight Control of a Quadrotor Vehicle Subsequent to a Rotor Failure”. In: *Journal of Guidance, Control, and Dynamics* 37.2 (2014), pp. 580–591. DOI: 10.2514/1.59869.

[66] V. Lippiello, F. Ruggiero, and D. Serra. “Emergency landing for a quadrotor in case of a propeller failure: A backstepping approach”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Sept. 2014, pp. 4782–4788. DOI: 10.1109/IROS.2014.6943242.

[67] C. de Crousaz, F. Farshidian, M. Neunert, and J. Buchli. “Unified motion control for dynamic quadrotor maneuvers demonstrated on slung load and rotor failure tasks”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. May 2015, pp. 2223–2229. DOI: 10.1109/ICRA.2015.7139493.

[68] J. Stephan, L. Schmitt, and W. Fichter. “Linear parameter-varying control for quadrotors in case of complete actuator loss”. In: *Journal of Guidance, Control, and Dynamics* 41 (10 2018), pp. 2232–2246. ISSN: 15333884. DOI: 10.2514/1.G003441.

[69] X. Wang, E.-J. van Kampen, Q.P. Chu, and P. Lu. “Stability Analysis for Incremental Nonlinear Dynamic Inversion Control”. In: *Journal of Guidance, Control, and Dynamics* 42 (5 May 2019), pp. 1116–1129. ISSN: 0731-5090. DOI: 10.2514/1.g003791.

[70] H. K. Khalil and J. W. Grizzle. *Nonlinear systems*. Vol. 3. Prentice Hall, 2002. ISBN: 9780132280242.

[71] V. Klein and E.A. Morelli. *Aircraft System Identification: Theory and Practice*. AIAA education series. American Institute of Aeronautics and Astronautics, 2006. ISBN: 9781563478321.

[72] E. A. Morelli. “Global nonlinear aerodynamic modeling using multivariate orthogonal functions”. In: *Journal of Aircraft* 32.2 (1995), pp. 270–277. DOI: 10.2514/3.46712.

[73] M. D. Zeiler and R. Fergus. “Visualizing and Understanding Convolutional Networks”. In: *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014, pp. 818–833. ISBN: 978-3-319-10590-1. DOI: 10.1007/978-3-319-10590-1_53.

[74] G. E. Hinton and R. R. Salakhutdinov. “Reducing the dimensionality of data with neural networks”. In: *Science* 313 (5786 July 2006), pp. 504–507. ISSN: 00368075. DOI: 10.1126/science.1127647.

[75] S. Lu and T. Başar. “Robust nonlinear system identification using neural-network models”. In: *IEEE Transactions on Neural Networks* 9 (3 1998), pp. 407–429. ISSN: 10459227. DOI: 10.1109/72.668883.

[76] S. Chen, S. A. Billings, and P. M. Grant. “Non-linear system identification using neural networks”. In: *International Journal of Control* 51 (6 1990), pp. 1191–1214. ISSN: 13665820. DOI: 10.1080/00207179008934126.

[77] K. Kirkpatrick, J. May, and J. Valasek. “Aircraft system identification using Artificial Neural Networks”. In: *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013*. American Institute of Aeronautics and Astronautics Inc., 2013. ISBN: 9781624101816. DOI: 10.2514/6.2013-878.

[78] J. Harris, F. Arthurs, J. V. Henrickson, and J. Valasek. “Aircraft system identification using artificial neural networks with flight test data”. In: *2016 International Conference on Unmanned Aircraft Systems, ICUAS 2016*. Institute of Electrical and Electronics Engineers Inc., June 2016, pp. 679–688. ISBN: 9781467393331. DOI: 10.1109/ICUAS.2016.7502624.

[79] S. A. Bagherzadeh. “Nonlinear aircraft system identification using artificial neural networks enhanced by empirical mode decomposition”. In: *Aerospace Science and Technology* 75 (Apr. 2018), pp. 155–171. ISSN: 12709638. DOI: 10.1016/j.ast.2018.01.004.

[80] R. Kumar, R. Ganguli, and S. N. Omkar. “Rotorcraft parameter estimation using radial basis function neural network”. In: *Applied Mathematics and Computation* 216 (2 Mar. 2010), pp. 584–597. ISSN: 00963003. DOI: 10.1016/j.amc.2010.01.081.

[81] A. Punjani and P. Abbeel. “Deep learning helicopter dynamics models”. In: *Proceedings - IEEE International Conference on Robotics and Automation*. Vol. 2015-June. Institute of Electrical and Electronics Engineers Inc., June 2015, pp. 3223–3230. DOI: 10.1109/ICRA.2015.7139643.

[82] A. J. Al-Mahasneh, S. G. Anavatu, and M. Garratt. “Nonlinear multi-input multi-output system identification using neuro-evolutionary methods for a quadcopter”. In: *9th International Conference on Advanced Computational Intelligence, ICACI 2017*. Institute of Electrical and Electronics Engineers Inc., July 2017, pp. 217–222. ISBN: 9781509047260. DOI: 10.1109/ICACI.2017.7974512.

[83] N. Mohajerin, M. Mozifian, and S. Waslander. “Deep Learning a Quadrotor Dynamic Model for Multi-Step Prediction”. In: *Proceedings - IEEE International Conference on Robotics and Automation*. Institute of Electrical and Electronics Engineers Inc., Sept. 2018, pp. 2454–2459. ISBN: 9781538630815. DOI: 10.1109/ICRA.2018.8460840.

[84] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of Control, Signals, and Systems* 2 (4 Dec. 1989), pp. 303–314. ISSN: 09324194. DOI: 10.1007/BF02551274.

[85] A. R. Barron. “Universal approximation bounds for superpositions of a sigmoidal function”. In: *IEEE Transactions on Information Theory* 39.3 (May 1993), pp. 930–945. ISSN: 1557-9654. DOI: 10.1109/18.256500.

[86] J. Heaton. *Introduction to Neural Networks with Java*. Heaton Research, 2008. ISBN: 9781604390087. URL: https://books.google.nl/books?id=Swlcw7M4uD8C.

[87] P. J. Werbos. “Backpropagation through time: what it does and how to do it”. In: *Proceedings of the IEEE* 78.10 (Oct. 1990), pp. 1550–1560. ISSN: 1558-2256. DOI: 10.1109/5.58337.

[88] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “ImageNet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60 (6 June 2017), pp. 84–90. ISSN: 15577317. DOI: 10.1145/3065386.

[89] Y. Lecun, Y. Bengio, and G. Hinton. “Deep learning”. In: *Nature* 521 (7553 May 2015), pp. 436–444. ISSN: 14764687. DOI: 10.1038/nature14539.

[90] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning representations by back-propagating errors”. In: *Nature* 323 (6088 Oct. 1986). ISSN: 0028-0836. DOI: 10.1038/323533a0.

[91] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[92] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[93] C. M. Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

[94] D. H. Wolpert. “Stacked generalization”. In: *Neural Networks* 5 (2 Jan. 1992), pp. 241–259. ISSN: 08936080. DOI: 10.1016/S0893-6080(05)80023-1.

[95] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. "Averaging Weights Leads to Wider Optima and Better Generalization". In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018* 2 (Mar. 2018), pp. 876–885. URL: `http://arxiv.org/abs/1803.05407`.

[96] G. Loianno, C. Brunner, G. McGrath, and V. Kumar. "Estimation, Control, and Planning for Aggressive Flight with a Small Quadrotor with a Single Camera and IMU". In: *IEEE Robotics and Automation Letters* 2 (2 Apr. 2017), pp. 404–411. ISSN: 23773766. DOI: `10.1109/LRA.2016.2633290`.

[97] Y. Li, S. Zahran, Y. Zhuang, Z. Gao, Y. Luo, Z. He, L. Pei, R. Chen, and N. El-Sheimy. "IMU/Magnetometer/Barometer/Mass-Flow Sensor Integrated Indoor Quadrotor UAV Localization with Robust Velocity Updates". In: *Remote Sensing* 11 (7 Apr. 2019), p. 838. ISSN: 2072-4292. DOI: `10.3390/rs11070838`.

[98] R. C. Leishman, J. C. Macdonald, R. W. Beard, and T. W. McLain. "Quadrotors and Accelerometers: State Estimation with an Improved Dynamic Model". In: *IEEE Control Systems Magazine* 34.1 (Feb. 2014), pp. 28–41. ISSN: 1941-000X. DOI: `10.1109/MCS.2013.2287362`.

[99] S. A. Ludwig and K. D. Burnham. "Comparison of Euler Estimate using Extended Kalman Filter, Madgwick and Mahony on Quadcopter Flight Data". In: *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018*. Institute of Electrical and Electronics Engineers Inc., Aug. 2018, pp. 1236–1241. ISBN: 9781538613535. DOI: `10.1109/ICUAS.2018.8453465`.

[100] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. "Vision-Based State Estimation and Trajectory Control Towards High-Speed Flight with a Quadrotor." In: *Robotics: Science and Systems*. Vol. 1. Citeseer. 2013, p. 32.

[101] F. Morbidi and G. Caron. "Phase correlation for dense visual compass from omnidirectional camera-robot images". In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 688–695. ISSN: 2377-3766. DOI: `10.1109/LRA.2017.2650150`.

[102] R. Mahony, T. Hamel, and J. Pflimlin. "Nonlinear Complementary Filters on the Special Orthogonal Group". In: *IEEE Transactions on Automatic Control* 53.5 (June 2008), pp. 1203–1218. ISSN: 1558-2523. DOI: `10.1109/TAC.2008.923738`.

[103] R. Mahony, T. Hamel, J. Trumpf, and C. Lageman. "Nonlinear attitude observers on SO(3) for complementary and compatible measurements: A theoretical study". In: *Proceedings of the IEEE Conference on Decision and Control*. 2009, pp. 6407–6412. ISBN: 9781424438716. DOI: `10.1109/CDC.2009.5399821`.

[104] S. O.H. Madgwick, A. J.L. Harrison, and R. Vaidyanathan. "Estimation of IMU and MARG orientation using a gradient descent algorithm". In: *IEEE International Conference on Rehabilitation Robotics*. 2011. ISBN: 9781424498628. DOI: `10.1109/ICORR.2011.5975346`.

[105] G. Welch and G. Bishop. *An Introduction to the Kalman Filter*. Tech. rep. USA, 1995.

[106] R. Anderson and D. M. Bevly. "Using GPS with a model-based estimator to estimate critical vehicle states". In: *Vehicle System Dynamics* 48.12 (2010), pp. 1413–1438. DOI: `10.1080/00423110903461347`.

[107] H. W. Sorenson. "Least-squares estimation: from Gauss to Kalman". In: *IEEE Spectrum* 7.7 (July 1970), pp. 63–68. ISSN: 1939-9340. DOI: `10.1109/MSPEC.1970.5213471`.

[108] S. Särkkä. *Bayesian filtering and smoothing*. Vol. 3. Cambridge University Press, 2013.

[109] P. Martin and E. Salaün. "Generalized multiplicative extended kalman filter for aided attitude and heading reference system". In: *AIAA Guidance, Navigation, and Control Conference*. 2010, p. 8300. DOI: `10.2514/6.2010-8300`.