Received: 20 December 2016 | Revised: 5 May 2017 | Accepted: 9 May 2017

DOI: 10.1002/rob.21729

REGULAR ARTICLE

WILEY

A framework for maximum likelihood parameter identification applied on MAVs

Michael Burri | Michael Bloesch | Zachary Taylor | Roland Siegwart | Juan Nieto

Autonomous Systems Lab, ETH Zurich, Switzerland

Correspondence
Michael Burri, Autonomous Systems Lab, ETH Zurich, Switzerland.
Email: burrimi@ethz.ch

Abstract

With the growing availability of agile and powerful micro aerial vehicles (MAVs), accurate modeling is becoming more important. Especially for highly dynamic flights, model-based estimation and control combined with a good simulation framework is key. While detailed models are available in the literature, measuring the model parameters can be a time-consuming task and requires access to special equipment or facilities. In this paper, we propose a principled approach to accurately estimate physical parameters based on a maximum likelihood (ML) estimation scheme. Unlike many current methods, we make direct use of both raw inertial measurement unit measurements and the rotor speeds of the MAV. We also estimate the spatial-temporal alignment to a modular pose sensor. The proposed ML-based approach finds the parameters that best explain the sensor readings and also provides an estimate of their uncertainty. Although we derive the proposed method for use with an MAV, the approach is kept general and can be extended to other sensors or flying platforms. Extensive evaluation on simulated data and on real-world experimental data demonstrates that the approach yields accurate estimates and exhibits a large region of convergence. Furthermore, we show that the estimation can be performed using only on-board sensing, requiring no external infrastructure.

1 | INTRODUCTION

Robotic systems often have a large number of physical parameters, which define their motion. These parameters may initially be unknown or can change over time. For instance, they can change when a new payload is added, an object is picked up, or simply due to wear and tear. Accounting for these changes is especially important on flying platforms where the parameters can have significant effects on the flight properties of the system. In this paper, we present a principled method for estimating the physical parameters of robotic systems and demonstrate the approach via its application to micro aerial vehicles (MAVs).

Accurate estimation of a wide range of parameters is important to allow accurate simulation of MAVs as well as correctly predicting a MAVs behavior given the systems inputs. This accurate behavior prediction is vital to highly dynamic control when the system makes use of a model-based controller. Furthermore, a good model enables the estimation of external disturbances on the MAV. Knowledge of these disturbances allows the controller to compensate for these in a feed-forward fashion reducing the reaction time and increasing the agility.

While some of the parameters (such as the systems mass) can be obtained by simple measurements, others (such as aerodynamic constants) can be challenging to measure, even when wind tunnel data are available. Because of this, it is highly desirable to have an approach that can obtain accurate estimates for the MAV parameters of interest from on-board sensor observations in an arbitrary environment, with no additional equipment. Owing to its utility, data-driven identification of aerial systems has been the focus of many researchers, especially on large-scale platforms.

In many applications, some parameters may change quickly, for instance the mass and inertia when an object is picked up. It is important to account for these changes directly in the controller to achieve good performance as shown in Mellinger et al.[1] One way of solving this problem is through an adaptive controller. By using an online estimation method such as an extended Kalman filter (EKF) to estimate the changing parameters, it is possible to directly update the control law.

A key drawback of online methods is that even a temporarily wrong estimate may have disastrous consequences on system behavior. There are many possibilities to minimize that risk. Most scenarios only require the estimation of a small subset of the parameters whereas other system parameters can be kept constant. This not only reduces the complexity of the estimation framework but more importantly minimizes the risk that the method may converge to an incorrect solution. Another approach to mitigate the risk of failures is through

the use of robust control or safety bounds on the adaptive controller. These can be provided in cases where the uncertainty of the parameters is available. Lastly, there are unobservable modes on MAVs, which require specific motions to render all parameters observable.

Many state-of-the-art approaches pose the identification as an off-line maximum likelihood (ML) problem, which can overcome many of the previously mentioned risks of online methods. ML optimization finds the underlying parameters for the system model that have the highest probability of generating the observed sensor readings over all data. Since the estimated parameters are not directly applied to the real system, they can be verified with user knowledge. Furthermore, as these methods make use of all available information without marginalizing out past data, they suffer from fewer problems from insufficient motion. Owing to the probabilistic nature of the approach, it is possible to include knowledge about the sensor noise and estimate the uncertainty of the parameters.

Successful system identification requires as much information as possible about the system dynamics and its time-varying state. However, in most cases it is not possible to measure the complete state directly. For MAVs, a pose measurement is especially helpful, since it provides rich information about the state. One way to measure this pose is through the use of external infrastructure such as the Global Positioning System (GPS) or a motion-tracking system. Another possibility that allows the system to be independent of any external infrastructure can be to use an on-board camera and inertial measurement unit (IMU) to estimate the pose as proposed in Bloesch et al.[2] In both cases, the exact location of these sensors is difficult to measure to the required precision. Additionally, these measurements are often provided with a different clock source, which creates uncertainty in the exact time that the measurements were taken. This problem is often referred to as extrinsic sensor calibration and widely studied in the area of IMU and camera calibration.[4]

In this paper, we show a framework that embeds any robotic system with unknown parameters into an ML estimation problem and apply it to a real MAV. By extending the MAV model with the extrinsic sensor calibration, minimal pre-processing of the measurements is necessary even in the case of unknown time offsets between the sensors. The framework is very general and can be extended to include other sensors and even other types of robotic systems.

The approach is validated on both simulations in an open source simulator RotorS[5] and on a real MAV using both off-board and entirely on-board pose sensing. Wind tunnel tests are also performed to demonstrate the suitability and accuracy of the method. This work is an extension to our previous conference paper[6] with significant enhancements and more thorough evaluation.

The main contributions of this work are

- Guidelines for how to transform any robotic system with unknown parameters into an ML estimation problem. Under certain assumptions, this allows the inclusion of all available sensor measurements in a statistically optimal way, enabling the estimation of the mean and covariance of the unknown parameters.
- The application of the method to MAV parameter identification, even in the case of unknown time delays (no pre-alignment needed).
- Thorough analysis and evaluation in simulation and real-world experiments, including ground truth (GT) data from a wind tunnel.
- Estimation of the spatial and temporal alignment of the pose sensor.
- Analysis of the estimated covariance.
- Comparison of the parameter estimates with an external tracking system and using only on-board data. This demonstrates independence of any external and expensive equipment.
- Practical hints for the initialization of the ML estimation with an extended Karman filter (EKF). We also perform a thorough analysis of the region of convergence to quantify the required accuracy of the initial guess for the unknown parameters.

The remainder of the paper is organized as follows. In Section 2, we present the related work with a focus on off-line methods. After introducing the notation in Section 3, we give an overview of our system in Section 4, and how this system can be identified using ML estimation in Section 5. Section 6 presents a detailed model of our MAV and in particular what the unknown state and parameters are. Finally, a thorough evaluation on simulated data as well as on a real MAV is presented in Section 7.

2 | RELATED WORK

Parameter identification is crucial to accurate modeling of MAVs, and a large amount of work has been performed toward solving this problem. Additionally, on a real system the intersensor calibration, often referred to as extrinsics, are important for state estimation and control. While we combine both issues into one problem, they have traditionally been solved as two separate subproblems. Because of this, we first focus on the work performed toward the identification of the dynamic model parameters of the MAV. Finally, we give a short overview of available visual inertial state estimation techniques that could be used for on-board pose estimation.

Much of the earlier work on MAVs used a combination of static tests and information from computer aided design (CAD) drawings for estimating the model parameters.[7] In Derafa et al.,[8] the inertia of the MAV is estimated using a pendulum test and the aerodynamic parameters are calculated from static thrust tests. Most of these experiments are easy to set up, but time consuming. We implemented similar methods to acquire GT in this work. However, more detailed aerodynamic models require the use of a wind tunnel.

For larger platforms, these tests are either impossible, not accurate enough, or hard to conduct. Because of this, significant effort was spent in the aerospace community on developing methods to estimate the system parameters. A survey of system identification publications for flying platforms is given in Hoffer et al.,[9] and we highlight only the most relevant for our application below. There are two main areas in system identification, online parameter estimation and off-line methods.

Early work on online parameter estimation was conducted in the frequency domain. Morelli[10] uses a recursive Fourier transformation, which essentially acts as a memory for all the measurement data in

the time domain. By only selecting a small band of interest in the frequency domain, the computational complexity remains low. Additionally, this allows the method to automatically reject noise in the system and bias effects. However, a common problem with frequency domain identification is the limitation to linear systems, which becomes especially problematic in the case of unknown rotations. Chowdhary and Jategaonkar[11] show a comparison of a EKF and unscented Kalman filter (UKF) for parameter estimation on a fixed wing aircraft and a unmanned aerial vehicle (UAV). Both estimators achieve similar results and converge to a reasonable solution.

In adaptive control, the updated parameters are directly used for control. This was, for instance, shown for inertia tracking in Chaturvedi and coworkers[12] for a generic 3 degrees of freedom (DoF) model. Rashid and Akhtar[13] adapted this approach and estimated the inertia of a multicopter. Adaptive control is particularly promising for aerial transportation, as shown in Mellinge et al.,[1] where the mass, center of gravity (CoG), and inertia of the MAV was estimated in a recursive estimator.

In the case of parameter estimation for MAVs, we believe that it is beneficial to use off-line methods and run the estimation over a full batch of data for two main reasons. First, in contrast to adaptive control it allows for the verification of the estimated parameters via the use of previous knowledge or other identification methods. This becomes especially important as the number of estimated parameters increases, as the risk of a wrong estimate becomes more probable. Second, some of the parameters are not easy to observe or have unobservable modes. This means that it is often difficult to excite the system in a way that renders all parameters observable at all times. Performing the estimation on more data increases the probability that there is a good motion present leading to a convergence of the unknown system parameters.

There is limited literature on MAV parameter identification as used in this work. However, there is a broad range of work for full-scale helicopters and planes[14,15] or using the same methods on a small size helicopter.[16] All these approaches use a frequency domain–based identification method, which is described in detail in Tischler and Remple.[14]

Jategaonkar[17] gives a good overview for parameter identification in the time domain. In this book, the most similar approach to our work is the filter error method (FEM), which is used in Jategaonkar and Plaetschke[18] to estimate the model parameters of a plane under process and measurement noise. Additionally, they describe how the noise densities can be identified automatically. An EKF is used to formulate the measurement residuals directly and only optimize over the unknown parameters, whereas in our approach also the unknown states are added to the optimization. This reduces the computational complexity at the cost of losing some information. The main problem is that in the FEM, only the current estimate is used for the linearization points, as opposed to our method, which uses all of the available information simultaneously for the selection of linearization points. Furthermore, it is unclear how this method could be extended to include all available measurements on our MAV.

One of the main requirements for a successful system identification is accurate measurements of the systems state. While this is less critical on large-scale aircrafts or helicopters that can carry heavy but highly accurate inertial navigation system (INS) systems, MAV are limited to lightweight sensors. In our case, this is a micro electro-mechanical systems (MEMS) IMU, which suffers from strong bias effects that have to be coestimated with the state of the system. Additionally, the position and orientation of the MAV is measured by an external tracking system (Vicon) or estimated using an on-board vision system such as in Bloesch et al.[2] This introduces two additional problems. First, the location of this pose sensor is usually not known exactly and can be hard to measure. Second, the time of the measurements is often with respect to another clock source than the on-board IMU of the MAV, and therefore an unknown time offset has to be estimated as well.

This intersensor calibration is the second part we integrated into our framework. In the early MAV literature, estimating the sensors location is referred to as self-calibration. Weiss et al.[19] estimate the transformation between a pose sensor, given by GPS or a visual-tracking algorithm with respect to the IMU using an EKF. This allows having a power on and go system without the need for a tedious calibration between the sensors, especially in the case where the location of the sensors changes slightly over time.

Considerable efforts in this problem have also been spend in the field of camera-IMU calibration, where we use the term extrinsics to refer to the offset between sensors. Kelly and Sukhatme[20] show the estimation for the camera IMU extrinsics using a UKF. In Li and Mourikis[21] not only the state and extrinsics between the camera and IMU are estimated, but the time offset between them is also tracked. We use a similar formulation to estimate the time delay in our framework. Similar to the system identification case, improved results are achieved in a batch ML estimation over all the states and unknown parameters. Fleps et al.[22] used B-splines to parametrize the state and show that the batch solution is superior to a UKF in terms of accuracy and more robust to wrong initialization. Furgale et al.[4] also used B-splines but extended it to also estimate the time delay between the sensors.

In our work, we make use of a formulation of the ML problem that is similar to the one proposed in Nikolic et al.,[23] where in addition to the extrinsic calibration of the camera and IMU, IMU-specific parameters such as the scale are estimated. Compared to the previous work with B-splines, the IMU is directly used as the motion model to predict the state in between measurements. This removes the necessity of using splines.

To be independent of any external infrastructure, cameras are usually used to estimate the pose on MAVs due to their low weight. For parameter identification, it is important to have aggressive motion and a high level of robustness is required. Therefore, we propose the use of a tightly coupled visual inertial odometry framework which uses the IMU directly for state estimation and significantly improves the robustness. Many of the available online methods are filtering based,[2,20,21] but there are also some optimization-based methods. These usually use a sliding window to keep the computational complexity low.[24] In our work, we used Bloesch et al.[2] due to the high robustness, low computational complexity, and open-source availability.

3 | PRELIMINARIES

Before we start formulating the parameter estimation problem, we want to give a brief overview of the notation and how we incorporate nonvector quantities (rotations) into our framework.

3.1 | Notation

Prescripts are used to specify the coordinate system in which a quantity is expressed. ${}_S\boldsymbol{r}_{BC}$ would refer to the vector from point B to C, expressed in coordinate system S. A rotation is given by Φ_{BS} with respect to a reference coordinate system S, and maps the coordinates of a vector ${}_S\boldsymbol{r}$ expressed in S to B:

$$ {}_B\boldsymbol{r} = \Phi_{BS}({}_S\boldsymbol{r}) \tag{1} $$

The same mapping can be expressed with a multiplication of the direction cosine matrix $\boldsymbol{C}(\Phi_{BS})$ and the vector ${}_S\boldsymbol{r}$

$$ {}_B\boldsymbol{r} = \boldsymbol{C}_{BS}\,{}_S\boldsymbol{r}, \tag{2} $$

where we used the shorthand notation $\boldsymbol{C}_{BS} = \boldsymbol{C}(\Phi_{BS})$. In our implementation, we use quaternions $\boldsymbol{q}(\Phi_{BS})$ to represent rotations and we make use of the same shorthand notation $\boldsymbol{q}_{BS} = \boldsymbol{q}(\Phi_{BS})$.

3.2 | Nonvector spaces

Some of the parameters we wish to estimate are defined on a nonvector manifold, and special care must be taken in the ML optimization. Mainly because manifolds only locally resemble a Euclidean space, addition and subtraction are not defined. In this work, the convention from Bloesch et al.[3] is used. Bloesch et al.[3] also derive several key identities; the most important of which are summarized below.

We use the left exponential map to generate a local orientation parametrization defined in the tangent space $\mathbb{R}^3$ (lie algebra) to the manifold $SO(3)$ (lie group). Similar to Hertzberg et al.[25] in Bloesch et al.,[3] the two operators $\boxplus$ and $\boxminus$ are defined to perform this mapping and replace the normal addition and subtraction known from vector spaces.

$$ \boxplus : SO(3) \times \mathbb{R}^3 \to SO(3), \tag{3} $$

$$ \Phi, \boldsymbol{\varphi} \mapsto \exp(\boldsymbol{\varphi}) \circ \Phi $$

$$ \boxminus : SO(3) \times SO(3) \to \mathbb{R}^3, \tag{4} $$

$$ \Phi_1, \Phi_2 \mapsto \log(\Phi_1 \circ \Phi_2^{-1}), $$

with a rotation vector $\boldsymbol{\varphi} \in \mathbb{R}^3$, which can be associated as the relative orientation of two coordinate systems.

For the optimization, we linearize around the nominal trajectory $\bar{\boldsymbol{x}}$ and use a local and minimal representation in the tangent space of the state vector $\boldsymbol{x}$ denoted with $\delta\boldsymbol{x}$. Measurements are indicated with a tilde ${}_B\tilde{\boldsymbol{\omega}}$, and estimated values are marked with a hat $\hat{\boldsymbol{x}}$.

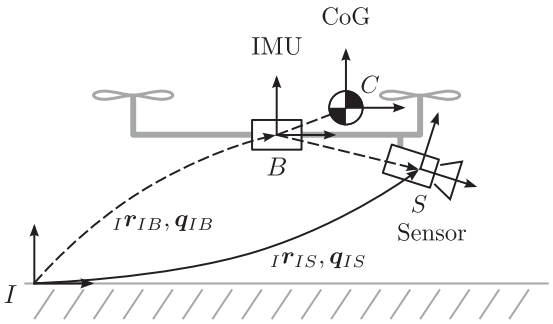

FIGURE 1 Main components and coordinate systems in this work, showing a rotary-wing MAV with an IMU and camera as the main sensors. Solid lines are measured quantities and dashed are estimated

4 | SYSTEM DESCRIPTION

Figure 1 shows the three main components of interest for this work, which are described in more detail in Section 6.

The first component is the MAV itself shown in gray and its CoG with corresponding coordinate frame C. In many cases, the exact location on the robot is unknown and most commonly assumed to be in the center of the MAV. The IMU's location is usually known and fixed on the robots body frame B. Our MAV's position is measured by a sensor S that measures position ${}_I\boldsymbol{r}_{IS}$ and orientation $\boldsymbol{q}_{IS}$ with respect to the world fixed inertial frame I. In this work, this pose is provided by either an external tracking system or estimated using the on-board camera. Again, this sensor is mounted at an unknown location and furthermore usually has an unknown time delay with respect to the IMU.

4.1 | MAV dynamic model description

The movement of a flying robot is defined by the Newton–Euler equations of motion around the CoG and can be summarized as

$$ \dot{\boldsymbol{x}} = \boldsymbol{f}_M(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{u}_M, \boldsymbol{w}_M). \tag{5} $$

This function $\boldsymbol{f}_M(\dots)$ depends on the state $\boldsymbol{x}$ of the system and evolves according to the actuator inputs $\boldsymbol{u}_M$. In our case, these are the rotor speeds, which are usually measured at a high rate with the same clock source as the IMU. The state includes the time-varying quantities of the system, such as position, velocity, and orientation. Owing to some unknown parameters like CoG location or aerodynamic coefficients, we additionally introduce an unknown parameter vector $\boldsymbol{\theta}$. Modeling uncertainties and external disturbances are incorporated into the dynamic models by means of a lumped noise term $\boldsymbol{w}_M \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_M)$.

4.2 | IMU model description

On most MAVs, there is an IMU which is an essential part for state estimation and control. This sensor provides acceleration $\tilde{\boldsymbol{a}}$ and angular rate measurements $\tilde{\boldsymbol{w}}$ at a high rate. There are two possible ways to integrate these measurements into our estimation framework, which are discussed in detail in Section 6.3.

In this work, the following differential IMU model $\boldsymbol{f}_I(\dots)$ is used:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_I(\boldsymbol{x}, \boldsymbol{u}_I, \boldsymbol{w}_I), \tag{6}$$

driven by the measurements $\boldsymbol{u}_I = [\tilde{\boldsymbol{a}}^T, \tilde{\boldsymbol{\omega}}^T]^T$ and continuous white noise $\boldsymbol{w}_I \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_I)$.

4.3 | Pose measurement description

To estimate the state of our system, we rely on a position and orientation (pose) measurements $\boldsymbol{z}$. Unfortunately, this pose is usually not reported in the body frame B and the offset is often challenging to measure. We therefore include this offset in our unknown parameter vector θ. This gives our measurement function $\boldsymbol{h}(\dots)$:

$$\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{x}, \theta, \boldsymbol{v}), \tag{7}$$

with a lumped noise term $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$.

Since this pose is provided by either an external tracking system or an on-board vision system that runs on a different clock source than the IMU, we must also estimate the offset between the clocks. This is especially important for the external tracking system, which runs on a different computer.

5 | METHOD

In formulating our approach, we had four main objectives:

1. provide a generic and flexible framework for parameter estimation of dynamic systems, such that it can be adapted to support a wide range of systems and sensors,
2. leverage all available information about the system provided by the sensors and system models,
3. use a probabilistic framework to incorporate both noise on the measurements and uncertainty due to model simplifications, and
4. provide an estimate of the uncertainty for all estimated parameters.

In our estimation framework, we are interested in finding the unknown parameter vector θ introduced in Section 4 and specified in (70). However, the parameters can only be estimated if the time-varying state $\boldsymbol{x}$ of the system is known, which generally cannot be directly measured. We therefore add the unknown state $\boldsymbol{x}_k = \boldsymbol{x}(t_k)$ to the estimation at discrete times t_k for all K measurements $\boldsymbol{z}_k = \boldsymbol{z}(t_k)$. This leads to the unknown vector $\boldsymbol{y} = [\boldsymbol{x}_0^T, \dots, \boldsymbol{x}_K^T, \theta^T]^T$, which we aim to estimate with our approach.

ML estimation is able to find the optimal $\boldsymbol{y}^*$, which maximize the probability of the observed measurements $\boldsymbol{z}$ and inputs $\boldsymbol{u}$:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}}{\operatorname{argmax}}\, p(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{y}), \tag{8}$$

where $\boldsymbol{z} = [\boldsymbol{z}_0^T, \boldsymbol{z}_1^T, \dots, \boldsymbol{z}_K^T]^T$ represents the measurements at time instances $k = 0 \dots K$ and $\boldsymbol{u} = [\boldsymbol{u}_0^T, \boldsymbol{u}_1^T, \dots, \boldsymbol{u}_N^T]^T$ contains all the N inputs to the dynamic systems.

In the case of a Gaussian distribution, the maximization of this probability density function is equivalent to minimizing the negative log likelihood function L, which is of the form

$$L(\boldsymbol{y}) = \sum_{i=0}^{K} \boldsymbol{r}_i(\boldsymbol{y})^T \boldsymbol{W}_i^{-1} \boldsymbol{r}_i(\boldsymbol{y}). \tag{9}$$

In this form, the available information is contained in a residual function $\boldsymbol{r}$ with an associated weight $\boldsymbol{W}$, which represents the uncertainty in the residual. This formulation gives the well-known nonlinear least squares (NLS) problem:

$$\boldsymbol{y}^* = \arg\min_{\boldsymbol{y}} L(\boldsymbol{y}). \tag{10}$$

Another common approach to solving such a problem is recursive estimation such as an EKF. The main advantage is a much lower computational complexity. However, our approach yields a higher accuracy as we optimize over all data at the same time and we iterate. This is especially helpful to reduce linearization errors. A good comparison with further detail can for instance be found in Strasdat et al.[26] From a practical side, the main problem is that Kalman filter (KF) based approaches are less flexible and need some additional tricks to include all information.[27]

In the following sections, we briefly summarize ML estimation, before defining the two types of residuals encountered in our system. After these definitions, we show how the presented theory can be applied to the MAV model.

5.1 | Nonlinear least squares estimation

The NLS problem can be solved iteratively. First, the likelihood function is approximated with a first-order Taylor expansion around the nominal trajectory $\bar{\boldsymbol{y}}$ (requires initial guess $\bar{\boldsymbol{y}}_0$). To simplify notation, we only show the iteration index n in the update (12).

$$\begin{aligned}
\bar{L}(\bar{\boldsymbol{y}} + \delta\boldsymbol{y}) &= \sum_{i=0}^{K} \boldsymbol{r}_i(\bar{\boldsymbol{y}} + \delta\boldsymbol{y})^T \boldsymbol{W}_i^{-1} \boldsymbol{r}_i(\bar{\boldsymbol{y}} + \delta\boldsymbol{y}) \\
&\approx \sum_{i=0}^{K} \left(\bar{\boldsymbol{r}}_i + \boldsymbol{J}_{y,i}\delta\boldsymbol{y}\right)^T \boldsymbol{W}_i^{-1} \left(\bar{\boldsymbol{r}}_i + \boldsymbol{J}_{y,i}\delta\boldsymbol{y}\right) \\
&= \sum_{i=0}^{K} \|\bar{\boldsymbol{r}}_i + \boldsymbol{J}_{y,i}\delta\boldsymbol{y}\|^2_{\boldsymbol{W}_i^{-1}}
\end{aligned} \tag{11}$$

This is the well-known weighted least squares problem, where the optimal error vector $\delta\boldsymbol{y}^*$ that minimizes $\bar{L}$ must be found. $\boldsymbol{J}_{y,i}$ is the Jacobian of the residual $\boldsymbol{r}_i$ with respect to the trajectory $\boldsymbol{y}$ and $\bar{\boldsymbol{r}}_i = \boldsymbol{r}_i(\bar{\boldsymbol{y}})$ the shorthand notation for the nominal residual.

The nominal trajectory can then be updated according to

$$\begin{aligned}
\delta\boldsymbol{y}^* &= \arg\min_{\delta\boldsymbol{y}} \sum_{i=0}^{K} \|\boldsymbol{r}_i(\bar{\boldsymbol{y}}_n) + \boldsymbol{J}_{y,i}\delta\boldsymbol{y}\|^2_{\boldsymbol{W}_i^{-1}} \\
\bar{\boldsymbol{y}}_{n+1} &= \bar{\boldsymbol{y}}_n + \delta\boldsymbol{y}^*.
\end{aligned} \tag{12}$$

Some of the parameters and part of the state vector involve orientations, which are members of $SO(3)$. Careful consideration of this manifold structure must be taken to ensure that these orientations are optimized in a correct manner. A detailed description with proofs is given in Bloesch et al.[3] Using the previous definitions, given in Section 3.2, we can rewrite (11) as

$$\begin{aligned}\bar{L}(\bar{\boldsymbol{y}} \boxplus \delta\boldsymbol{y}) &= \sum_{i=0}^{K} \boldsymbol{r}_i(\bar{\boldsymbol{y}} \boxplus \delta\boldsymbol{y})^T \boldsymbol{W}_i^{-1} \boldsymbol{r}_i(\bar{\boldsymbol{y}} \boxplus \delta\boldsymbol{y}) \\ &\approx \sum_{i=0}^{K} \|\bar{\boldsymbol{r}}_i + \boldsymbol{J}_{y,i}\delta\boldsymbol{y}\|^2_{\boldsymbol{W}_i^{-1}} \qquad (13) \\ &= \|\boldsymbol{b}(\bar{\boldsymbol{y}}) + \boldsymbol{A}\delta\boldsymbol{y}\|^2. \qquad (14)\end{aligned}$$

We assume that the residuals are defined in a minimal vector space (i.e., in the tangent space for nonvector spaces). This is important, as it allows for the removal of the $\boxplus$ operator in the linearized likelihood function (13). In the last step, all the residuals are stacked into a single residual vector $\boldsymbol{b}(\bar{\boldsymbol{y}}) = \boldsymbol{W}^{-T/2}\boldsymbol{r}(\bar{\boldsymbol{y}})$ and scaled accordingly. This results in the classical least squares problem. The same holds for the scaled Jacobian matrix $\boldsymbol{A} = \boldsymbol{W}^{-T/2}\boldsymbol{J}_y$.

The nominal trajectory then follows an update that is similar to (12)

$$\bar{\boldsymbol{y}}_{n+1} = \bar{\boldsymbol{y}}_n \boxplus \delta\boldsymbol{y}^*. \qquad (15)$$

Note, that for most of the optimization variables (i.e., position), which already lie in a vector space $\mathbb{R}^3$, the operator $\boxplus$ is equivalent to standard addition.

5.2 | Covariance recovery

The covariance of all states and parameters can be recovered by inverting the information matrix given by $(\boldsymbol{A}^T\boldsymbol{A}) = \boldsymbol{\Sigma}^{-1}$ with the Jacobian matrix $\boldsymbol{A}$ defined in (14). For long data sets the information matrix can be very large and the inversion takes a long time. However, for in-field calibration, the result should be available in a short time. In this work, we are only interested in the uncertainty of the parameters and instead of the full inversion just solve a smaller subproblem.

We make use of the approach presented in Kaess et al.[28] to only recover the last section of the covariance matrix. This is done via a forward and back-substitution and significantly decreases computation time.

Since the ordering of our optimization variables $\boldsymbol{y}$ can be chosen freely, we use the last entries for the parameter vector. In that case, the covariance matrix $\boldsymbol{\Sigma}$ has the following structure:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{\theta x}^T \\ \boldsymbol{\Sigma}_{\theta x} & \boldsymbol{\Sigma}_{\theta\theta} \end{bmatrix}. \qquad (16)$$

5.3 | Measurement residuals

For dynamic systems two types of residuals are possible. The first type of residual only affect a single state $\boldsymbol{x}_k$ at the time the measurement was taken t_k; in our case, these form the pose measurements (7). Owing to this relationship, we therefore need to add a state $\boldsymbol{x}_k$ at every time instance t_k a measurement is received. Furthermore, these measurements sometimes depend on some unknown parameter vector $\boldsymbol{\theta}$. We start by defining the measurement residuals $\boldsymbol{r}_{z,k}$ and then derive the weighting according to the uncertainty.

The residual $\boldsymbol{r}_{z,k}$ we want to minimize is given by the difference between the measurement $\boldsymbol{z}_k$ and a function of the state $\boldsymbol{h}(\dots)$

$$\boldsymbol{r}_{z,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{v}_k) = \boldsymbol{z}_k \boxminus \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{v}_k) \qquad (17)$$

where the $\boxminus$ is used to account for orientations as was discussed in Section 3.2 and Gaussian noise $\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$. It is also possible to use an implicit measurement function that directly depends on the measurement. The only requirement for this is that the residual is defined in a vector space (in the tangent space for orientations).

$$\boldsymbol{r}_{z,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{v}_k) = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{z}_k, \boldsymbol{v}_k) \qquad (18)$$

As in the general case of NLS, we perform a Taylor expansion of the measurement residual.

$$\boldsymbol{r}_{z,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{v}_k) \approx \bar{\boldsymbol{r}}_{z,k} + \boldsymbol{H}_{x,k}\delta\boldsymbol{x}_k + \boldsymbol{H}_{\theta,k}\delta\boldsymbol{\theta}_k + \boldsymbol{H}_{v,k}\boldsymbol{v}_k \qquad (19)$$

The small error vectors $\delta\boldsymbol{x} = \boldsymbol{x} \boxminus \bar{\boldsymbol{x}}$ and $\delta\boldsymbol{\theta} = \boldsymbol{\theta} \boxminus \bar{\boldsymbol{\theta}}$ are now vector quantities in the tangent space. $\bar{\boldsymbol{r}}_{z,k} = \boldsymbol{r}_{z,k}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0)$ is the residual evaluated at the nominal trajectory and the Jacobians are given by the partial derivatives:

$$\boldsymbol{H}_{x,k} = \frac{\partial \boldsymbol{r}_z(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{z}_k, \boldsymbol{v}_k)}{\partial \boldsymbol{x}_k}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, \boldsymbol{z}_k, 0) \qquad (20)$$

$$\boldsymbol{H}_{\theta,k} = \frac{\partial \boldsymbol{r}_z(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{z}_k, \boldsymbol{v}_k)}{\partial \boldsymbol{\theta}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, \boldsymbol{z}_k, 0) \qquad (21)$$

$$\boldsymbol{H}_{v,k} = \frac{\partial \boldsymbol{r}_z(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{z}_k, \boldsymbol{v}_k)}{\partial \boldsymbol{v}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, \boldsymbol{z}_k, 0). \qquad (22)$$

We define the partial derivative with respect to variables on the manifold and refer the reader to Bloesch et al.[3] for a more detailed discussion.

The weighting of this residual is given by the probability of the residual

$$p(\boldsymbol{r}_{z,k}|\boldsymbol{x}_k, \boldsymbol{\theta}) \sim \mathcal{N}(0, \overbrace{\boldsymbol{H}_{v,k}\boldsymbol{R}\boldsymbol{H}_{v,k}^T}^{\boldsymbol{W}_{z,k}}) \qquad (23)$$

Note that the weight of the residual $\boldsymbol{W}_{z,k}$ must be full rank for the NLS optimization, as it must be invertible.

5.4 | Differential residuals

The second type of residuals in dynamic systems arise from differential equations $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{w})$. In our application, these equations are the MAV dynamic model defined in (5) or IMU model (6).

The residual for the model $\boldsymbol{r}_{M,k}$ is the difference between the state $\boldsymbol{x}_{k+1}$ and its prediction from the model $\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{f}_{d,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)$. $\boldsymbol{f}_{d,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)$ is the discrete function, which integrates the differential equation given by the model from time t_k to t_{k+1}. Owing to this relationship, these residuals always depend on two states.

A Taylor expansion is applied to these residuals in a manner similar to those discussed previously. After this, the residuals are given by

$$\boldsymbol{r}_{M,k}(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k) = \boldsymbol{x}_{k+1} \boxminus \boldsymbol{f}_{d,k}(\boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k) \tag{24}$$

$$\approx \bar{\boldsymbol{r}}_{M,k} + \boldsymbol{A}_{k+1}\delta\boldsymbol{x}_{k+1} + \boldsymbol{F}_k\delta\boldsymbol{x}_k + \boldsymbol{F}_{\theta,k}\delta\boldsymbol{\theta} + \boldsymbol{F}_{w,k}\boldsymbol{w}_k. \tag{25}$$

These residuals are evaluated at the nominal trajectory with zero noise $\bar{\boldsymbol{r}}_{M,k} = \boldsymbol{r}_{M,k}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0)$, and the Jacobians are given by the partial derivatives evaluated at the nominal trajectory.

$$\boldsymbol{A}_{k+1} = \frac{\partial \boldsymbol{r}_{M,k}(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)}{\partial \boldsymbol{x}_{k+1}}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) \tag{26}$$

$$\boldsymbol{F}_k = \frac{\partial \boldsymbol{r}_{M,k}(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)}{\partial \boldsymbol{x}_k}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) \tag{27}$$

$$\boldsymbol{F}_{\theta,k} = \frac{\partial \boldsymbol{r}_{M,k}(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)}{\partial \boldsymbol{\theta}}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) \tag{28}$$

$$\boldsymbol{F}_{v,k} = \frac{\partial \boldsymbol{r}_{M,k}(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k, \boldsymbol{\theta}, \boldsymbol{w}_k)}{\partial \boldsymbol{w}_k}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) \tag{29}$$

Previously defined measurement residuals are fixed in time and cannot be shifted. Therefore, we use the differential residuals as a generic way to express the predicted state $\hat{\boldsymbol{x}}_{k+1}$ at time t_{k+1} as a function of the state $\boldsymbol{x}_k$. Furthermore, there are many cases where there are multiple small integration steps for the model and calculating $\boldsymbol{f}_{d,k}$ becomes slightly more involved. Examples of situations in which this can occur are high-rate IMU or motor speed measurements and time shifts due to asynchronous sensors. There is a large amount of literature covering this topic from simple first-order approximations[29] to preintegration on the manifolds[30] to avoid repeating this costly step every iteration.

An example of asynchronous measurements is shown in Figure 2.

We introduce a local version of all the states $\boldsymbol{x}'_i$, Jacobians and time t'_i within one residual to avoid multiple indexes. The time runs from $t'_0 = t_k$ through $t'_L = t_{k+1}$ with L timesteps.

We solved this integration by splitting the problem into shorter integrals and use a fourth-order Runge–Kutta method to calculate the state $\hat{\boldsymbol{x}}'_{i+1}$.

$$\hat{\boldsymbol{x}}'_{i+1} = \overbrace{\int_{t'_i}^{t'_{i+1}} \boldsymbol{f}_M(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{u}_i, \boldsymbol{w}_i)dt}^{\boldsymbol{f}'_{d,i}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{w}_i)}, \tag{30}$$

where we initialize the first state of the integration $\boldsymbol{x}'_0 = \bar{\boldsymbol{x}}_k$ and repeat this process until we reach $\hat{\boldsymbol{x}}'_L = \hat{\boldsymbol{x}}_{k+1}$ at time t_{k+1}. The function $\boldsymbol{f}_{d,k}$ is then given by a concatenation of the integrated functions $\boldsymbol{f}'_{d,i}$

$$\boldsymbol{f}_{d,k} = \boldsymbol{f}'_{d,L-1} \circ \dots \circ \boldsymbol{f}'_{d,i} \circ \dots \circ \boldsymbol{f}'_{d,0}. \tag{31}$$

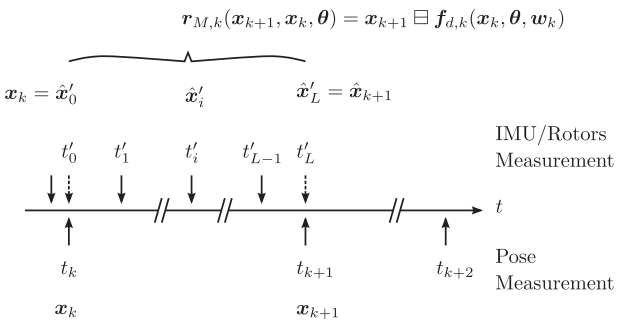

FIGURE 2 Two consecutive states given by the pose measurements are connected by a differential constraint. To simplify notation, we introduce a local time t' for the residual which runs from $t'_0 = t_k$ to $t'_L = t_{k+1}$. We introduce a interpolated measurement shown with a dotted line at t'_0 and t'_L. By integrating the differential equations, we can combine the intermediate states $\hat{\boldsymbol{x}}'_i$ to one constraint and calculate the Jacobians and weighting of the residual

For calculating the Jacobians and the weighting of the residuals, the function $\boldsymbol{f}'_{d,i}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{w}_i)$ is approximated with a first-order (Euler) method. The probability of the integrated state is then given by

$$p(\hat{\boldsymbol{x}}'_{i+1}|\hat{\boldsymbol{x}}'_i, \boldsymbol{\theta}) \sim \mathcal{N}(\hat{\boldsymbol{x}}'_{i+1}, \overbrace{\boldsymbol{F}'_i\boldsymbol{P}'_i\boldsymbol{F}'^{\mathsf{T}}_i + \boldsymbol{G}'_i\boldsymbol{Q}'_i\boldsymbol{G}'^{\mathsf{T}}_i}^{\boldsymbol{P}'_{i+1}}), \tag{32}$$

where $\boldsymbol{Q}'_i$ is the covariance of the integrated continuous time noise. At the beginning of the integration at time t'_0, there is no uncertainty in the first state $\boldsymbol{P}'_0 = \boldsymbol{0}$.

The Jacobians $\boldsymbol{F}'_i, \boldsymbol{G}'_i$ are given by the partial derivatives:

$$\boldsymbol{F}'_i = \frac{\partial \boldsymbol{f}'_{d,i}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{w}_i)}{\partial \boldsymbol{x}_i}(\hat{\boldsymbol{x}}'_i, \bar{\boldsymbol{\theta}}, 0), \tag{33}$$

$$\boldsymbol{G}'_i = \frac{\partial \boldsymbol{f}'_{d,i}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{w}_i)}{\partial \boldsymbol{w}_i}(\hat{\boldsymbol{x}}'_i, \bar{\boldsymbol{\theta}}, 0). \tag{34}$$

By using the chain rule, the Jacobians of the integrated residual can now be calculated:

$$\boldsymbol{A}_{k+1} = \frac{\partial \boldsymbol{x}_{k+1} \boxminus \boldsymbol{f}_{d,k}(\dots)}{\partial \boldsymbol{x}_{k+1}}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) \tag{35}$$

$$\boldsymbol{F}_k = \overbrace{\frac{\partial \boldsymbol{x}_{k+1} \boxminus \boldsymbol{f}_{d,k}(\dots)}{\partial \boldsymbol{f}_{d,k}(\dots)}}^{\boldsymbol{J}_{F,k}} \frac{\partial \boldsymbol{f}_{d,k}(\dots)}{\partial \boldsymbol{x}_k}(\bar{\boldsymbol{x}}_{k+1}, \bar{\boldsymbol{x}}_k, \bar{\boldsymbol{\theta}}, 0) = \boldsymbol{J}_{F,k}\boldsymbol{F}'_L \dots \boldsymbol{F}'_0 \tag{36}$$

For the fixed parameter Jacobian, we get

$$\boldsymbol{F}_{\theta,k} = \boldsymbol{J}_{F,k}(\boldsymbol{F}'_{\theta,L} + \boldsymbol{F}'_L(\boldsymbol{F}'_{\theta,L-1} + \boldsymbol{F}'_{L-1}(\dots))), \tag{37}$$

which is best solved during the integration. The weighting of the integrated residual $\boldsymbol{W}_{M,k}$ is given by the integrated covariance matrix $\boldsymbol{W}_{M,k} = \boldsymbol{P}'_L$.

Note that for vector spaces $\boldsymbol{A}_{k+1}$ is simply the identity matrix and $\boldsymbol{J}_{F,k}$ the matrix's negative identity. Similarly, in case of small

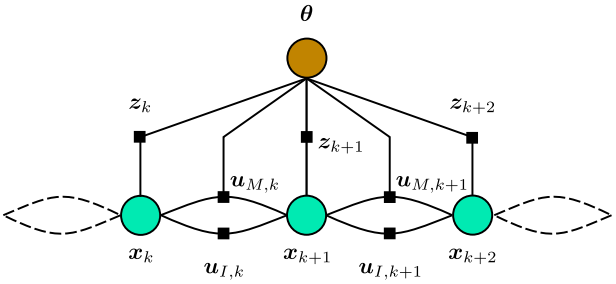

FIGURE 3 Graphical illustration of our identification problem in form of a factor graph. The nodes of the graph are the unknown parameters $\boldsymbol{\theta}$ and states $\boldsymbol{x}_k$ we want to estimate. Every pose measurement $\boldsymbol{z}_k$ depends on one state and the parameters, defining the first type of factor. By integrating the differential equation given by the MAV dynamics and IMU dynamic model, we can formulate the second type of factor, which connects two states

rotational errors these matrices are close to identity and are often neglected.

5.5 | MAV parameter estimation with NLS

Figure 3 illustrates our formulation of the problem in the form of a factor graph. In our estimation framework, we want to estimate the unknown parameter vector $\boldsymbol{\theta}$ and the states $\boldsymbol{x}_k$ which are represented with a circle. These unknown values are connected by the two different types of residuals, indicated with a square and referred to as factors. The pose measurements $\boldsymbol{z}_k$ are fixed in time only depending on one state $\boldsymbol{x}_k$ and on the unknown parameters, defining our measurement residuals $\boldsymbol{r}_{z,k}$. Note that for every pose measurement, a state must be added to the problem.

The second type of factor is given by differential equations and depends on two states. In our identification, one of them is the MAV dynamic model given by the rotor speeds $\boldsymbol{u}_M$, which depends on the unknown parameters. The other is the IMU model, which is independent of the parameters and given by the measurements $\boldsymbol{u}_I$. As shown in Section 5.4, the differential equation are integrated from time t_k to t_{k+1} to connect the two corresponding states $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$. In the case of multiple motor speed measurements or IMU measurements between the two states, these inputs are stacked into one input vector $\boldsymbol{u}_{M,k}$, respective $\boldsymbol{u}_{I,k}$. This allows us to transform a complex system with varying rates and delays into the simple structure shown in Figure 3. Using these definitions, we are able to form our differential residuals $\boldsymbol{r}_{M,k}(\boldsymbol{x},\boldsymbol{\theta})$ and $\boldsymbol{r}_{I,k}(\boldsymbol{x},\boldsymbol{\theta})$.

The ML estimate of our unknown parameters and states are now found by minimizing the following likelihood function.

$$\begin{aligned}\bar{\mathrm{L}}(\boldsymbol{x},\boldsymbol{\theta}) = &\sum_{k=0}^{K} \boldsymbol{r}_{z,k}(\boldsymbol{x},\boldsymbol{\theta})^T \boldsymbol{W}_{z,k}^{-1} \boldsymbol{r}_{z,k}(\boldsymbol{x},\boldsymbol{\theta}) && \text{Measurements}\\ &+ \sum_{k=1}^{K} \boldsymbol{r}_{M,k}(\boldsymbol{x},\boldsymbol{\theta})^T \boldsymbol{W}_{M,k}^{-1} \boldsymbol{r}_{M,k}(\boldsymbol{x},\boldsymbol{\theta}) && \text{MAV model}\\ &+ \sum_{k=1}^{K} \boldsymbol{r}_{I,k}(\boldsymbol{x},\boldsymbol{\theta})^T \boldsymbol{W}_{I,k}^{-1} \boldsymbol{r}_{I,k}(\boldsymbol{x},\boldsymbol{\theta}). && \text{IMU} \quad (38)\end{aligned}$$

In our work, we made use of the Gauss–Newton method to solve this optimization problem. This method was chosen due to its ease of implementation and was empirically found to give good results in our application. However, many other possible solvers exist that can exhibit superior convergence properties, such as the Levenberg–Marquardt approach.

5.6 | Estimator initialization

For the maximum likelihood estimation, we need an initial guess of the state trajectory $\bar{\boldsymbol{x}}$ to select a good linearization point. This initial trajectory can be estimated with an EKF. Since we wish to be independent of the MAV-specific parameters, only the IMU model and pose measurements are used for the initialization. This leads to the common IMU and pose measurement fusion described in, for example, Lynen et al.[31]

We still need to initialize the extrinsic parameters of the pose sensor and the IMU. While the translation between the IMU and the pose sensors is not as critical and can be assumed close to zero, a good initial guess is needed for the rotation. To form this initial estimate, we first numerically calculate the angular rate of the orientation measurement. This is then compared to the angular rate measured by the IMU in the algorithm proposed inKabsch[32] to give the initial guess for the rotation. In our experience, this was found to work very reliably. Furthermore, we also estimate the time offset between the pose sensor and the IMU, which is implemented in a manner similar to the work presented in Li and Mourikis.[21] This initialization technique leads to a highly generic algorithm that can handle raw measurements from any pose source, even in the presence of large time delays and rotational as well as translational offsets.

A summary of the approach we developed is presented in Algorithm 1 and closely resembles our implementation. In the InitializeEstimator() we first calculate a coarse alignment of the pose sensor to the IMU and then run an EKF to initialize the nominal trajectory $\bar{\boldsymbol{y}}$ which consists of the unknown states and parameters. Then we iteratively solve the ML problem with $n_{\max}$ iterations. In EvaluateResiduals() the residuals described in Section 5.5 are evaluated and stacked into the weighted residual vector $\boldsymbol{b}$ and weighted Jacobian matrix $\boldsymbol{A}$. We then solve the least squares problem using Gauss–Newton in SolveLeastSquaresProblem(). Once the iterations are finished, we extract the estimated parameters, which are the last entries of $\bar{\boldsymbol{y}}$ and recover the covariance which is described in Section 5.2.

ALGORITHM 1 Maximum likelihood parameter identification

$n := 0$
$\bar{\boldsymbol{y}} :=$ InitializeEstimator ()
% Solve ML problem
while $n < n_{max}$ **do**
  $\boldsymbol{b}, \boldsymbol{A} :=$ EvaluateResiduals $(\bar{\boldsymbol{y}})$
  $\delta\boldsymbol{y} :=$ SolveLeastSquaresProblem $(\boldsymbol{b}, \boldsymbol{A})$
  $\bar{\boldsymbol{y}} = \bar{\boldsymbol{y}} \boxplus \delta\boldsymbol{y}$
$\boldsymbol{\theta}^* :=$ ExtractParameters $(\bar{\boldsymbol{y}})$
$\boldsymbol{\Sigma}_\theta :=$ RecoverParameterCovariance $(\boldsymbol{A})$
return $\boldsymbol{\theta}^*, \boldsymbol{\Sigma}_\theta$

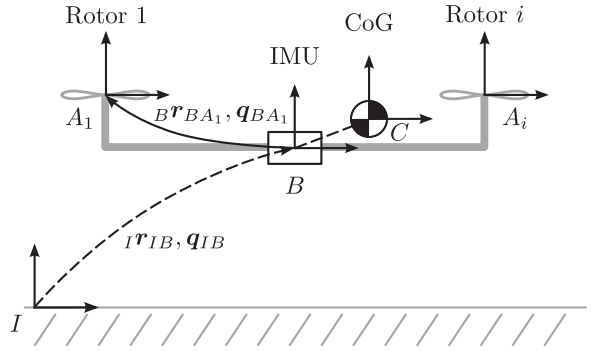

FIGURE 4 A schematically depicted MAV showing the body frame B, the CoG frame C, and the rotor frames $A_1 \dots A_i$

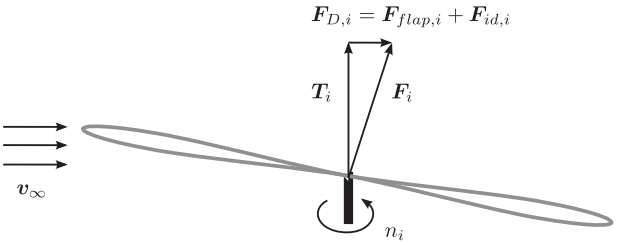

FIGURE 5 A schematic MAV rotor and the acting forces on it. The rotor plane is slightly tilted as a result of blade flapping, which is caused by different tip velocities for advancing and retreating blades relative to the air velocity. Under the assumption of no wind, this velocity is identical to the velocity of the rotor hub $\boldsymbol{v}_{hub,i}$

6 | SYSTEM MODELING

A wide range of different MAV models are available in existing literature, ranging from simplistic point mass models to extremely complex models of the aerodynamics.[33] For our work, we use a quite simple and commonly used model similar to the ones described in Mahony et al.[34] and Omari et al.[35] The motion of the MAV is mainly defined by the forces $\boldsymbol{F}_i$ and torque $\boldsymbol{M}_i$ that every rotor i produces on the main body at the rotor hub frame A_i, as shown in Figure 4. Therefore, we start by summarizing the forces and moments on a single rotor, before formulating the MAV dynamic model used for the parameter identification. After which, the IMU model and pose measurements are defined. We close the modeling section with the definition of the state and parameter vector we want to optimize.

6.1 | Forces and moments on a single rotor

We model two forces on each rotor i shown in Figure 5. The main force is the thrust force $\boldsymbol{T}_i$, which always points along the z-axis of the rotor $\boldsymbol{e}_z$. In addition to this force, there is the perpendicular drag force $\boldsymbol{F}_{D,i}$. This leads to the combined force $\boldsymbol{F}_i$ expressed in the rotor-attached frame A_i:

$$\boldsymbol{F}_i = \boldsymbol{T}_i + \boldsymbol{F}_{D,i} + \boldsymbol{w}_F, \tag{39}$$

where we add normally distributed process noise $\boldsymbol{w}_F$ to account for aerodynamic uncertainties. Since these quantities are all expressed locally in the rotor-attached frame A_i, we simplify the notation by dropping the frame of reference.

The thrust force T_i is proportional to the squared rotor velocity n_i

$$T_i = c_T n_i^2; \quad \boldsymbol{T}_i = T_i \boldsymbol{e}_z. \tag{40}$$

where c_T is the positive rotor thrust coefficient.

This coefficient strongly depends on the strength and angle of the relative wind speed (free air stream) $\boldsymbol{v}_\infty$, as has been suggested in the literature[36] and confirmed in our own wind tunnel experiments in Section 7.1. Therefore, we model the thrust coefficient as a random walk with white process noise w_{c_T}.

$$\dot{c}_T = w_{c_T} \tag{41}$$

According to the literature, there are two main effects responsible for the drag force, namely blade flapping and induced drag. Blade flapping is a phenomenon where the rotor plane of a MAV tilts during flight as shown in Figure 5. It comes about, as in forward flight the rotor tip velocity relative to the free stream $\boldsymbol{v}_\infty$ is different for the advancing blade and the retreating blade. This creates a force imbalance that leads to a tilting of the blades or "flapping."

For a detailed analysis of this phenomenon, we refer the reader to Mahony et al.[34] and Omari et al.,[35] with only the simplified result for the generated blade flapping force $F_{\text{flap},i}$ stated here.

$$\boldsymbol{F}_{\text{flap},i} = T_i \boldsymbol{A}_{\text{flap}} \boldsymbol{v}_\infty; \quad \boldsymbol{A}_{\text{flap}} = \begin{bmatrix} c_a & -c_b & 0 \\ c_b & c_a & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{42}$$

where c_a, c_b are the longitudinal and lateral flapping coefficient.

Induced drag is a result of the down wash, which changes the direction of the inflow velocity to the rotor. In hovering, this inflow is balanced out; but in forward flight, the advancing blade generates a larger induced drag force than the retreating. The resulting induced drag force is given by $\boldsymbol{F}_{id,i} = T_i \,\text{diag}(c_{id}, c_{id}, 0)\boldsymbol{v}_\infty$.

Under the assumption that blade flapping is mainly acting in the direction of the inflow velocity, c_b can be taken to be small. This results in a lumped drag coefficient $c_D = c_a + c_{id}$ and a drag force $\boldsymbol{F}_{D,i}$ that is given by

$$\boldsymbol{F}_{D,i} = T_i \begin{bmatrix} c_D & 0 & 0 \\ 0 & c_D & 0 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{v}_\infty. \tag{43}$$

It is important to note that the rotor drag force, as it is introduced above, increases in a manner that is linearly proportional to the free stream velocity.

In the case of no wind, the free stream velocity is identical to the negative rotor hub velocity, which is given by

$$\boldsymbol{v}_\infty = -\boldsymbol{C}_{BA_i}^T ({}_B\boldsymbol{v} + {}_B\boldsymbol{\omega} \times {}_C\boldsymbol{r}_{CA_i}), \tag{44}$$

where ${}_C\boldsymbol{r}_{CA_i}$ denotes the offset from the rotor hub to the CoG, ${}_B\boldsymbol{v}$ the velocity of the MAV, and ${}_B\boldsymbol{\omega}$ the angular rates, both expressed in body coordinates. For most MAVs, the rotors are not tilted and the rotation $\boldsymbol{C}_{BA_i}$ of the rotor plane with respect to the body frame is simply identity.

We only model one moment on the individual rotor, which is caused by the rotor drag. Similar to the thrust, this moment is proportional to the squared motor speed n_i and the positive rotor moment coefficient c_m.

$$\boldsymbol{M}_i = c_m n_i^2 \boldsymbol{e}_z + \boldsymbol{w}_M \tag{45}$$

To account for aerodynamic uncertainties, normally distributed process noise $\boldsymbol{w}_M$ is added to the model.

6.1.1 | Total force and moment

The total force is the sum of the forces over all N rotors and is given by

$${}_B\boldsymbol{F}_{tot} = \sum_{i=1}^{N} \boldsymbol{C}_{BA_i} {}_{A_i}\boldsymbol{F}_i. \tag{46}$$

For the total force, we neglect the fuselage drag, which would be proportional to the square of the velocity. This is valid for the relatively slow velocities used in our identification as will be shown in the wind tunnel experiments.

Similarly, the total moment around the CoG is given by

$${}_C\boldsymbol{M}_{tot} = \sum_{i=1}^{N} \boldsymbol{C}_{BA_i} \boldsymbol{M}_i + {}_B\boldsymbol{r}_{CA_i} \times {}_B\boldsymbol{F}_i. \tag{47}$$

Since we have no rotation between the B and C frame, the forces are the same in the body and the CoG frame, i.e., ${}_C\boldsymbol{F}_i = {}_B\boldsymbol{F}_i$. The effective lever arm is given by ${}_B\boldsymbol{r}_{CA_i} = {}_B\boldsymbol{r}_{BA_i} - {}_B\boldsymbol{r}_{BC}$ and is visualized in Figure 4.

6.2 | MAV dynamic model

We assume that the location of the IMU with respect to the rotors is known and does not change. Our state is expressed in the IMU B frame, which highly simplifies including the IMU measurements. Since the angular rate is the same everywhere on a rigid body, we neglect the frame of reference for angular velocities to simplify the notation.

Departing from most of the current work, we drop the assumption that the IMU is in the CoG of the MAV, as shown in Figure 4. In this case, the equation of motion is given by

$${}_B\dot{\boldsymbol{v}}_B = \frac{1}{m} {}_B\boldsymbol{F}_{tot} - \boldsymbol{C}_{IB}^T \cdot {}_I\boldsymbol{g} - \boldsymbol{\omega} \times {}_B\boldsymbol{v}_B \tag{48}$$

$$- \dot{\boldsymbol{\omega}} \times {}_B\boldsymbol{r}_{BC} - \boldsymbol{\omega} \times \boldsymbol{\omega} \times {}_B\boldsymbol{r}_{BC}, \tag{49}$$

where m is the mass, ${}_B\boldsymbol{v}_B$ the velocity, and $\boldsymbol{C}_{IB}$ the orientation of the MAV. The gravitational acceleration ${}_I\boldsymbol{g} = [\,0 \quad 0 \quad g\,]^T$ is expressed in the inertial frame and needs to be transformed into the B frame.

The rotational dynamics are given by

$$\dot{\boldsymbol{\omega}} = {}_C\boldsymbol{J}^{-1} \left({}_C\boldsymbol{M}_{tot} - \boldsymbol{\omega} \times {}_C\boldsymbol{J}\boldsymbol{\omega} \right), \tag{50}$$

with the inertia matrix ${}_C\boldsymbol{J}$. Note that care must be taken with the total moments ${}_C\boldsymbol{M}_{tot}$ and the inertia matrix, which are not independent of the frame they are expressed in. Since a flying body rotates around the CoG, these quantities are expressed in the C frame.

We now have all the definitions required to write down the explicit MAV dynamic model $\dot{\boldsymbol{x}} = \boldsymbol{f}_M(\boldsymbol{x}, \theta, \boldsymbol{u}_M, \boldsymbol{w}_M)$ introduced in Section 4.1.

$${}_I\dot{\boldsymbol{r}}_{IB} = \boldsymbol{C}_{IB} {}_B\boldsymbol{v}_B \tag{51}$$

$${}_B\dot{\boldsymbol{v}}_B = \frac{1}{m} {}_B\boldsymbol{F}_{tot} - \boldsymbol{C}_{IB}^T {}_I\boldsymbol{g} - \boldsymbol{\omega} \times {}_B\boldsymbol{v}_B \tag{52}$$

$$- \dot{\boldsymbol{\omega}} \times {}_B\boldsymbol{r}_{BC} - \boldsymbol{\omega} \times \boldsymbol{\omega} \times {}_B\boldsymbol{r}_{BC} \tag{53}$$

$$\dot{\Phi}_{IB} = -{}_I\boldsymbol{\omega}_{BI} = \boldsymbol{C}_{IB}\,\boldsymbol{\omega}_{IB}, \tag{54}$$

$$\dot{\boldsymbol{\omega}} = {}_C\boldsymbol{J}^{-1} \left({}_C\boldsymbol{M}_{tot} - \boldsymbol{\omega} \times {}_C\boldsymbol{J}\boldsymbol{\omega} \right) \tag{55}$$

$$\dot{c}_T = w_{c_T} \tag{56}$$

6.3 | IMU Model

The on-board MEMS IMU is comprised of an accelerometer and a rate gyroscope. They provide measurements of the MAV's acceleration $\tilde{\boldsymbol{a}}$ and angular velocity $\tilde{\boldsymbol{\omega}}$ at a high update rate. We assume that the inertial measurements are corrupted by white noise and a slowly varying bias process:

$$\tilde{\boldsymbol{a}} = \boldsymbol{a} + \boldsymbol{b}_a + \boldsymbol{w}_a \tag{57}$$

$$\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \boldsymbol{b}_w + \boldsymbol{w}_w \tag{58}$$

Random walk processes $\boldsymbol{b}_a$ and $\boldsymbol{b}_w$ with diffusion $\sigma_{ba}\boldsymbol{I}$ and $\sigma_{bw}\boldsymbol{I}$ model the accelerometer and gyroscope bias processes. $\boldsymbol{w}_a$ and $\boldsymbol{w}_w$ are Gaussian white noise processes of strength $\sigma_a^2\boldsymbol{I}$ and $\sigma_w^2\boldsymbol{I}$.

There are two possible ways to integrate these measurements into our estimation framework. The first way is to model the measurement $\boldsymbol{z}_I$ with a function $\boldsymbol{h}(\dots)$ that directly depends on the state $\boldsymbol{x}$ with some Gaussian noise $\boldsymbol{v}_I$, that is,

$$\boldsymbol{z}_I = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{v}_I). \tag{59}$$

This requires that the accelerations and angular rates are part of the state, not only increasing the state size but more importantly requiring the inclusion of a state in the optimization for every single IMU measurement.

The second approach is to employ a measurement model that depends on two subsequent state. By employing a numerical differentiation scheme, this allows for the removal of one differentiation level from the filter state and the combination of multiple measurements into a single residual, as shown in Section 5.4. In the context of Kalman filtering, this is analogous to a process model which propagates one state to another while taking into account the IMU measurements $\boldsymbol{u}_I = [\tilde{\boldsymbol{a}}^T, \tilde{\boldsymbol{\omega}}^T]^T$.[20] This leads to the differential IMU model $\dot{\boldsymbol{x}} = f_I(\boldsymbol{x}, \boldsymbol{u}_I, \boldsymbol{w}_I)$ introduced in Section 4.2. The detailed model is given by

$${}_B\dot{\boldsymbol{v}}_B = \tilde{\boldsymbol{a}} - \boldsymbol{b}_a - \boldsymbol{w}_a - \boldsymbol{C}_{IB}^T {}_I\boldsymbol{g} \tag{60}$$

$$-(\tilde{\boldsymbol{\omega}} - \boldsymbol{b}_w - \boldsymbol{w}_w) \times {}_B\boldsymbol{v}_B \tag{61}$$

$$\dot{\Phi}_{IB} = \Phi_{IB}(\tilde{\boldsymbol{\omega}} - \boldsymbol{b}_w - \boldsymbol{w}_w), \tag{62}$$

$$\dot{\boldsymbol{b}}_w = \boldsymbol{w}_{b_w} \tag{63}$$

$$\dot{\boldsymbol{b}}_a = \boldsymbol{w}_{b_a}, \tag{64}$$

FIGURE 6 A wind tunnel was used to measure the aerodynamic properties of our experimental MAV. Here we see the experimental setup for measuring the main body drag F_A

where $\boldsymbol{w}_{b_w}$ and $\boldsymbol{w}_{b_a}$ denote the white noise processes that drive the gyroscope and accelerometer bias variations (see (57) and (58)).

6.4 | Pose sensor model

In this paper, we assume that position and orientation (pose) measurements are available to the system. For our evaluation, we used either the on-board pose estimate given by our visual inertial odometry algorithm ROVIO[2] or a Vicon* motion-tracking system. Unfortunately, the pose is usually timestamped with a different clock source and it is important to compensate for this clock offset. With a slight abuse of notation, we define the clock source of reference in a similar way to the coordinate system a vector is expressed in. The time t_k with respect to the IMU clock B is then given by ${}_Bt_k$, and the time offset between the sensor S and B is given by t_{BS}.

Similar to a coordinate transformation, we can now do the same change of reference clock for timestamps.

$$ {}_Bt_k = t_{BS} + {}_St_k. \tag{65} $$

Similar to the IMU bias, this time difference is modeled as a random walk process and we define the following dynamic system:

$$ \dot{t}_{BS} = w_{t_{BS}}, \tag{66} $$

with white noise $w_{t_{BS}}$.

Since we defined the state and dynamic models of the MAV and IMU with respect to the IMU B, we need to transform the pose measurements. The measurement $\boldsymbol{z} = h(\boldsymbol{x}, \theta, \boldsymbol{v})$ introduced in Section 4.3 is then given by

$$ {}_I\tilde{\boldsymbol{r}}_{IS,k} = {}_I\boldsymbol{r}_{IB}(t_{BS} + {}_St_k) + \boldsymbol{C}_{IB}\,{}_B\boldsymbol{r}_{BS} + \boldsymbol{w}_p \tag{67} $$

$$ \tilde{\Phi}_{IS,k} = \Phi_{IB}(t_{BS} + {}_St_k) \circ \Phi_{BS} \boxplus \boldsymbol{w}_\phi, \tag{68} $$

where we correct for the time offset and depend on the state at the sensor measurement time ${}_St_k$.

$\boldsymbol{w}_p$ denotes position measurement errors, modeled as a discrete white Gaussian noise process of strength $\boldsymbol{R}_p$, $\boldsymbol{w}_p \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_p)$. w_ϕ describes attitude measurement errors with normally distributed error angle axis, with $\boldsymbol{w}_\phi \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_q)$.

6.5 | State and parameter definition

For the optimization, we have to distinguish between time-varying quantities, such as the vehicle position, and fixed parameters. The time-varying quantities are incorporated into the system state $\boldsymbol{x}(t)$, whereas the fixed parameters are summarized in a parameter vector θ. Our system state is defined as follows:

$$ \boldsymbol{x} = \left[{}_I\boldsymbol{r}_{IB}^T,\ {}_B\boldsymbol{v}_B^T,\ \boldsymbol{q}_{IB}^T,\ {}_B\boldsymbol{\omega}^T,\ \boldsymbol{b}_w^T,\ \boldsymbol{b}_a^T,\ c_T,\ t_{BS} \right]^T. \tag{69} $$

Note that in our implementation we used quaternions to represent orientations and the transpose here refers to the quaternion vector.

The unknown MAV model parameters and sensor extrinsics are

$$ \theta = \left[c_M,\ c_D,\ {}_C\boldsymbol{j}^T,\ {}_B\boldsymbol{r}_{BC}^T,\ {}_B\boldsymbol{r}_{BS}^T,\ \boldsymbol{q}_{BS}^T \right]^T. \tag{70} $$

The inertia matrix ${}_C\boldsymbol{J}$ is assumed diagonal, with diagonal elements ${}_C\boldsymbol{j}^T = [J_{xx}, J_{yy}, J_{zz}]$.

With these definitions, the dynamic equations from Sections 6.2 and 6.3; and measurement definition in Section 6.4: we have all the required information to formulate the ML optimization shown in Section 5.5.

7 | EXPERIMENTS

Our experiments were conducted with an AscTec Firefly (Fig. 6), a research MAV with six rotors that has been equipped with a visual inertial sensor.[37] We first conducted experiments in a wind tunnel to

* http://www.vicon.com/products/camera-systems/bonita

gather GT for the aerodynamic coefficients. Furthermore, we use the data of these experiments to justify some modeling decisions and highlight the need for a simpler parameter identification routine. Because many of the parameters are hard to measure, a first set of identification experiments is performed in simulation. We tried to simulate the real MAV as accurately as possible and used similar noise values for the real and simulated platform.

Finally, the core part of our experiments is evaluated on real-world data recorded on our MAV. We show that the identification can cope well with the noisy measurements of the low-cost MEMS IMU of the on-board autopilot. So we are evaluating our framework under the lowest SNR conditions, in terms of sensing. One of the parameters depends on the inflow velocity which is strongly influenced by the wind. For this reason, we propose to do the identification indoor as it is difficult to model external wind disturbances present in outdoor scenarios accurately. In this work, all the flights were performed in an indoor office-like setting with no artificial markers or modifications to the environment that would benefit the visual odometry. However, to allow for accurate odometery estimates there should be sufficient light and texture present. The identification is first performed using pose estimation provided by an external tracking system and then entirely from on-board vision-based state estimation. We show that the optimization converges even in the presence of large initial error on the parameters and give some practical hints as to how to initialize these.

7.1 | Ground truth experiments

The first set of experiments was intended to estimate the aerodynamic coefficients. A single propeller was mounted on a force torque sensor under 5 different tilting angles. We recorded data of six wind velocities $\boldsymbol{v}_\infty$ using the same control sequence for the motor. The results for the estimated aerodynamic coefficients are shown in the upper part of Figure 7. Each data point is the result of a LS fit, where the brightness corresponds to the tilting angle between the rotor plane and the wind velocity $\boldsymbol{v}_\infty$. We started with a tilting angle of 0° (black) and went up to 60° (light gray).

As suggested by momentum theory,[38] the thrust coefficient c_T strongly depends on the inflow velocity and tilting angle of the rotor blade. This effect can already be observed at low speeds. In the ML identification experiments, we therefore added this coefficient to the state vector and estimated it for every time sample. The other two coefficients c_M and c_D remained constant at low speeds and are therefore fixed parameters in our identification.

Although the test stand was equipped with some damping elements between the motor and sensor, the measurements were extremely noisy. This can be seen in the lower part of Figure 7, where the experimental data for one identification run is shown. Additionally, in some cases there were some resonance effects and a slowly varying bias on the force torque sensor due to the high electrical currents close to the sensor. This further highlights the need for a different identification approach, such as our ML method, which is based only on recorded flight data.

Figure 6 shows the experimental setup for measuring the main body drag F_A. The MAV was mounted on a force torque sensor, and the force

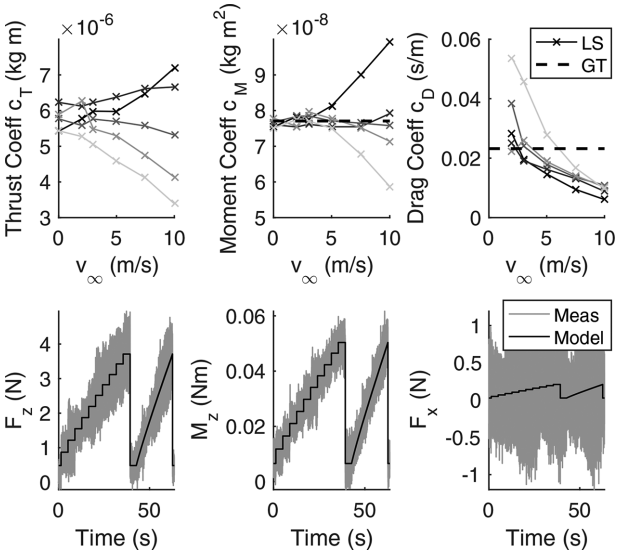

FIGURE 7 The upper section shows the aerodynamic coefficients found in the wind tunnel experiments, with five different tilting angles and varying wind speeds $\boldsymbol{v}_\infty$. Each cross marks the result of a least squares fit between the measured data and the model. We started with 0° tilting angle shown in black (rotor plane aligned with wind) and went up to 60° where brightness indicates bigger angles. The thrust coefficient c_T is highly dependent on the angle and wind speed and, therefore, is not assumed to be constant in our identification experiments. We instead use a random walk model for the thrust coefficient and add it to the state vector. Experimental data for one identification run is shown in the bottom row. Unfortunately, there was significant noise present in the measurements and a changing bias on the force torque sensor

was measured at seven different wind velocities. We used the estimated drag coefficient c_D from the previous experiment and hovering thrust to predict the drag force F_D. Unfortunately, the minimum wind speed possible in the wind tunnel was 2 m/s, which is approximately the nominal flight speed in our experimental trajectories. At these low speeds, the main body drag is small compared to the drag force F_D as can be seen in Figure 8. Therefore, the main body drag was not estimated in the following identification results.

Besides the wind tunnel experiments, we used the following simple methods to acquire GT. Estimating the inertia of our MAV was done with a pendulum test.[39] We measured the distance of the point of rotation (on one of the rotor arms) and the CoG and recorded the angular rate with the IMU to calculate the period. The inertia can then be calculated using the parallel axes theorem.

Ground truth for the mass offset ${}_B\boldsymbol{r}_{BC}$ was retrieved from CAD data. Since the MAV is well balanced, we assume that the CoG is approximately in the center. We therefore take the location of the IMU with respect to the center of the physical frame derived from the CAD model as GT.

7.2 | Quantitative analysis in simulation

For a quantitative evaluation with exact ground truth (GT), we rely on simulation data. We generated informative identification trajectories

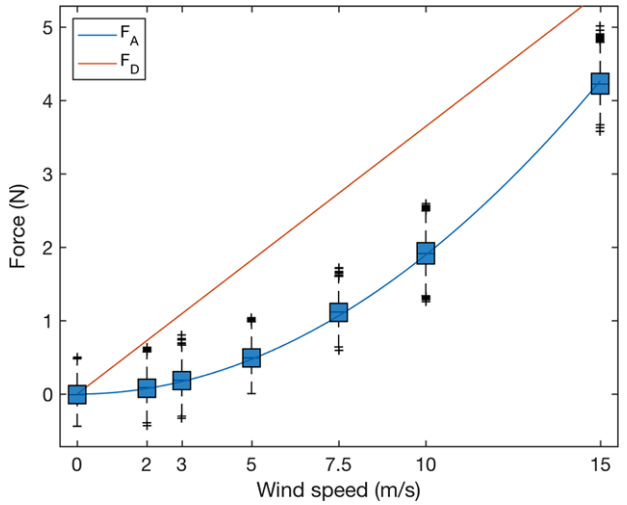

FIGURE 8 Main body drag F_A found in the wind tunnel experiments and the drag force F_D produced by the propeller in hovering. In our experiments, we were always at speeds below 3 m/s and therefore neglected the main body drag

that are 30 s in length using planning in belief space.[40] Every trajectory was stored and flown three times using the RotorS simulator.[5] The same trajectories were also used on the real platform experiments.

The simulator is implemented using Gazebo* and the open dynamics engine† for the physics. For a fair comparison with the real experimental platform, we extended the MAV model in RotorS to match the real helicopter as close as possible using similar noise values on the virtual sensors. Owing to the underlying physics engine, the simulated model is more complete than the one used for identification, which includes some approximations.

The mean and standard deviation over 15 simulated trajectories are summarized and compared against GT in Table 1. All parameters converge within a few Gauss–Newton iterations, even in the presence of initial error in the parameters. There are no significant errors between the mean over all estimates and the GT, suggesting that our model matches the RotorS model well. As expected, the standard deviation of the inertia around the z-axis is much bigger than the inertia around the other axes because of the strong coupling with the moment coefficient c_M. These two parameters are only distinguishable because of a small coupling term $\boldsymbol{\omega} \times {}_C\boldsymbol{J}\boldsymbol{\omega}$ in (50). Another interesting effect is that the CoG offset converges with low uncertainty, except for the z-axis. The reason is that a small offset in x and y direction has a large impact on the produced moment and is integrated quickly to a wrong attitude. Meanwhile, an offset in z-axis has a smaller effect on the overall motion of the MAV.

7.3 | Real-world experiments

On the physical MAV, we ran and recorded the same 30-s long trajectories as in the simulation experiments. With these experiments, we wanted to verify and critically reflect on the following claims:

* http://gazebosim.org/

† http://www.ode.org/

TABLE 1 We used the RotorS Gazebo simulator on 15 simulated trajectories. In all experiments the parameters settle close to their true values. The standard deviation of the error between the GT and estimated values is also small with three exceptions (marked in bold). However, these exceptions are consistent with the real platform. We get a larger spread in the moment constant c_M and inertia around the z-axis J_{zz} due to a strong coupling of these parameters. The CoG in z-axis is also difficult to estimate due to the small influence on the MAV motion

Parameter	Unit	Ground Truth	Estimated Value Mean	Error StDev
c_M	kg m^2	1.37e-07	1.4e-07	**3.82e-09**
c_D	s m^{-1}	0.0176	0.0161	0.000728
J_{xx}	kg m^2	0.0358	0.0373	0.0003
J_{yy}	kg m^2	0.0469	0.0483	0.000226
J_{zz}	kg m^2	0.101	0.102	**0.00232**
$r_{BC,x}$	mm	6.45e-05	0.0611	0.088
$r_{BC,y}$	mm	−0.000129	0.00931	0.0648
$r_{BC,z}$	mm	0.859	2.83	**1.51**
$r_{BS,x}$	mm	10	10.4	1.15
$r_{BS,y}$	mm	−20	−20.7	1.74
$r_{BS,z}$	mm	30	27.7	1.2
$roll_{BS}$	°	5.73	5.73	0.036
$pitch_{BS}$	°	−11.5	−11.5	0.0438
yaw_{BS}	°	17.2	17.2	0.0346

- The proposed approach works on the real MAV and converges to the correct solution.
- Estimation of the sensor to IMU transformation as well as time offset between the two is possible, removing the need for any preprocessing.
- The probabilistic nature of the approach allows us to calculate a meaningful uncertainty, together with the mean of the parameters.
- We can estimate the parameters using only on-board data, demonstrating independence from any external and expensive equipment.
- The approach is easy to use and converges to the correct solution even when poor initialization values are used. To show this, we conducted a small parameter study to determine the region of attraction and give some further practical hints on the initialization of the parameters.

For most of the experiments, we used pose measurements from an external tracking system (Vicon). However, we also perform the parameter estimation using only on-board sensing in Section 7.3.4. We used ROVIO[2] instead of Vicon to estimate the pose based on camera images and IMU data.

7.3.1 | Parameter convergence

In Figure 9, a small selection of parameters is shown for all 15 recorded flights. Many of the missing parameters are shown in later sections. We used an EKF to estimate the initial trajectory $\bar{\boldsymbol{x}}$ and then performed 10 iterations of Gauss–Newton. Using this approach the parameters quickly converge to physically meaningful values and no longer change

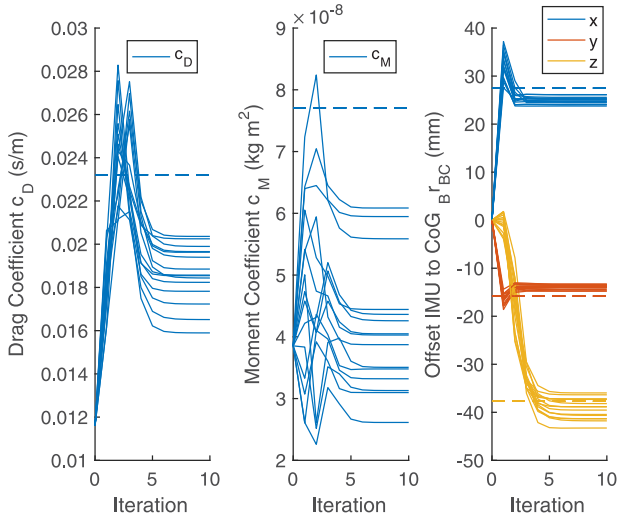

FIGURE 9 A small selection of parameters from 15 different trajectories. The GT values are indicated with a dashed line. Iteration 0 shows the initial guess followed by the result of 10 Gauss–Newton optimization steps. The convergence is quick, and the parameters no longer change significantly after only five iterations. With the lone exception of the moment coefficient c_M, the values are close to the GT

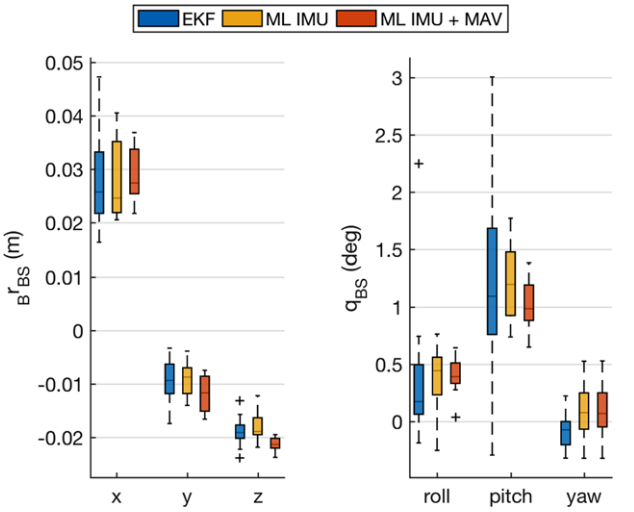

FIGURE 10 The offset in position ${}_B\boldsymbol{r}_{SB}$ and orientation q_{BS} between the pose sensor and the IMU is one of the most critical parameters for state estimation. This parameter can be estimated without the MAV dynamic model using an EKF or batch ML estimation over all IMU data. Here we show the comparison of the parameters after convergence over the 15 flown trajectories. As expected, the estimated parameters show a smaller spread in their values as we add more information

significantly after only five iterations. This is especially true for the CoG, which can be estimated with high certainty leading to millimeter accuracy. Similar to the simulation results, the z-axis has a larger spread than the other two axis.

The most difficult parameter to estimate is the moment coefficient c_M. As expected, this value has a large error due to the strong coupling with the inertia J_{zz} in the z-direction. However, there is a significant offset to the GT, which was not observed in the simulation results. We assume this offset is caused by some unmodeled aerodynamic effects on the real MAV. In contrast, to our static wind tunnel experiments, during flights we rarely have a stationary system. The aerodynamic properties are hard to model in these dynamic cases, which may be the cause of the lower moment coefficient in our recorded data sets. However, because most state-of-the-art model based controllers use the same simplified model we are using for the ML estimation, having this offset is not an issue.

7.3.2 | Comparison of different methods

We included the transformation between the pose sensor and the IMU in our unknown parameter vector. This parameter can be estimated without inclusion of the MAV dynamic model, and we utilize this for the following two comparisons: The first is the comparison of our ML estimation to a filtering approach. Second, we want to evaluate the benefits of having a joint optimization over all data, rather than a separate prealignment step used in previous work.

We evaluate the accuracy to which this offset is estimated by first comparing the estimated position ${}_B\boldsymbol{r}_{BS}$ and orientation $\boldsymbol{q}_{BS}$ offset over the 15 trajectories in Figure 10. The first estimate is given by a filtering approach. We implemented an EKF that uses only pose and IMU information, and we use the result of this filter to initialize the ML estimation. Here we only consider the last estimated state at the end of the trajectory, this ensures that the filter has sufficient time to converge.

The second estimate is given by a batch ML estimation using the same data as the EKF, i.e., pose sensor and IMU only. Compared to the EKF, the results are less spread for the orientation, but similar for the position. One important observation we made during the development of our framework is that the EKF depends much more on a correct initialization than the ML estimation. This is expected as the EKF does not iterate and is much more influenced by linearization errors. But as long as the initial state is close to the actual state, the performance is good.

Finally, we make use of all of the available information and add the MAV dynamic model in the approach shown in red. This further increases the consistency of the estimates. All three approaches show good results with millimeter accuracy for position and subdegree accuracy for the orientation.

Since external pose sensing or on-board pose estimation run on a different clock source than the MAV microcontroller, it turns out to be beneficial to also estimate the clock drift t_{BS}. The resulting clock drift for the first two trajectories are visualized in Figure 11 and is in the order of a few milliseconds. Although the EKF shown in blue converges toward the end of the data sets, the ML batch optimization refines the time offset substantially and gives a good estimate from the beginning of the trajectory.

7.3.3 | Estimated parameter uncertainty

One of the main benefits of ML parameter estimation is that an estimate of the parameter uncertainty can be recovered. Thanks to the probabilistic framework, once the sensor models are known, no further hand tuning is needed. However, recovering the covariance for all states is very expensive to calculate and should be avoided. Owing to a clever ordering of the variables, only a small part of the information matrix with the dimension of the unknown parameter vector needs to be inverted as outlined in Section 5.2. This allows for quick feedback on

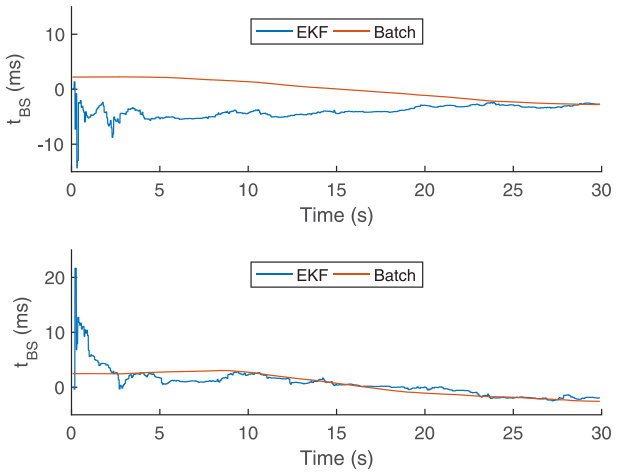

FIGURE 11 Time offset estimation between the pose sensor and the IMU for the first two trajectories. The EKF converges to similar values as the ML estimate toward the end in most cases. Inherent to the batch approach the ML optimization already provides a valid estimate at the beginning of the trajectory

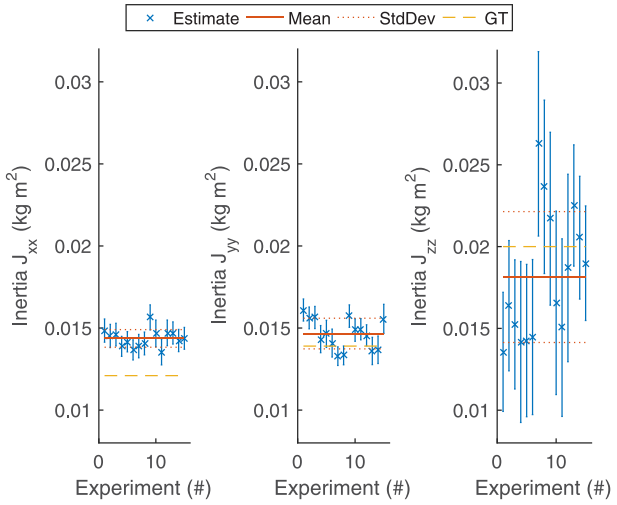

FIGURE 12 Estimated mean and standard deviation (1 σ) over all recorded trajectories for the inertia. Furthermore the overall mean, standard deviation, and GT is shown for comparison. Although the number of experiments is not large, the uncertainties seem to match well. Similar to the simulation, we have a bigger uncertainty on the inertia in the z-axis J_{zz}

the mean and covariance of the parameters in a short time (30 s with the sparse Matlab* implementation and 10 iterations).

To verify that our estimated covariance resembles a physically meaningful number, we show the estimated inertia parameters and the corresponding one sigma bound in Figure 12. Furthermore, we show the GT value and mean over all identification trajectories, as well as the 1, σ bounds of the calculated standard deviation from the 15 runs. As expected, the estimates in I_z are less certain than in the other two axes, but overall the estimated standard deviation seems to resemble the spread in our experiments quite well. The accuracy of these parameters do however depend on the accuracy of the process and measurement noise of the underlying models.

The estimation in the x-axis does not closely match our measured inertia that well, and the inertia in the y-direction is also slightly larger than expected. There are two possible explanations for these dependencies. One explanation is that it is hard to accurately measure the inertia in our GT experiments. However, another possible explanation is that our MAV has the IMU, battery, sensors, and computer connected with silicon dampers to the actual frame. These dampers are not evenly spaced, allowing the damped parts to move more freely in the x-axis than the y-axis. This has a similar effect to a larger inertia J_{xx}.

7.3.4 | On-board tracking

For in-field calibration, it is important to be independent of any external infrastructure. We propose to do the identification in an indoor environment to avoid wind disturbances, since wind is hard to model accurately. In these experiments, we used an office-like setting with a flight area of approximately 4.6 × 5 × 2.6 m. No artificial markers or other navigation aids were placed in the environment. This space should be available in most situations making this approach applicable for in-field calibration in cases where changes occur to the platform. In our experience, the lighting conditions and texture available in such environments are well suited for visual odometry.

We use Bloesch et al.[2] to estimate the pose of the MAV and directly replace the Vicon measurements. This approach is particularly well suited as it can cope well with fast motions and the resulting motion blur. These motions are required to excite all modes of the system and improve the quality of the parameter identification. Furthermore, it is a very robust approach and we experienced no failures in all data sets.

Owing to the extrinsic estimation shown in Section 7.3.2, the only adaptation needed was to slightly increase the noise on the provided pose. One common problem with visual odometry approaches is that they accumulate some error over time. However, since we do not have any parameters that depend on the absolute position, this is not an issue and the identification procedure could even work only using velocity estimates.

The differences between these experiments and the previous ones are quite small for most of the parameters. Because of this, we only show a comparison of the estimated inertia, which showed the most significant changes. In Figure 13, it can be seen that we have a larger spread along the y-axis in our 15 data sets. This seems to be due to a problem with the last three flights, where ROVIO experienced much larger pose errors than in the previous experiments, leading to an incorrect estimate. However, on the z-axis the estimates are more consistent and even closer to our measured GT.

Overall the estimated parameters are close to the previous estimates, where we used an external tracking system.

7.3.5 | Region of attraction

Inherent to the ML estimation is the need for an initial guess of the unknown variables. The proposed method is only useful if it converges

* http://www.mathworks.com/

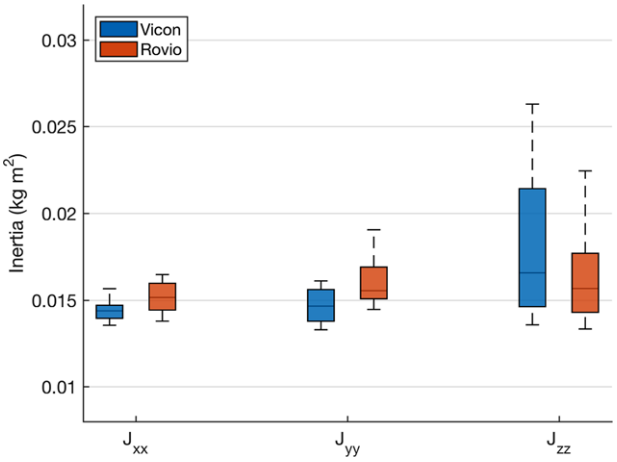

FIGURE 13 We also performed the parameter estimation with pose estimates provided by the visual-inertial algorithm ROVIO. Compared to the estimates provided with Vicon, the quality of the estimates is comparable for all parameters, showing that our method even works without expensive external infrastructure. Only the estimate of the inertia in the *y*-direction performed slightly worse. We think the reason for this is a problem with ROVIO, which experienced significant drift in position in these data sets

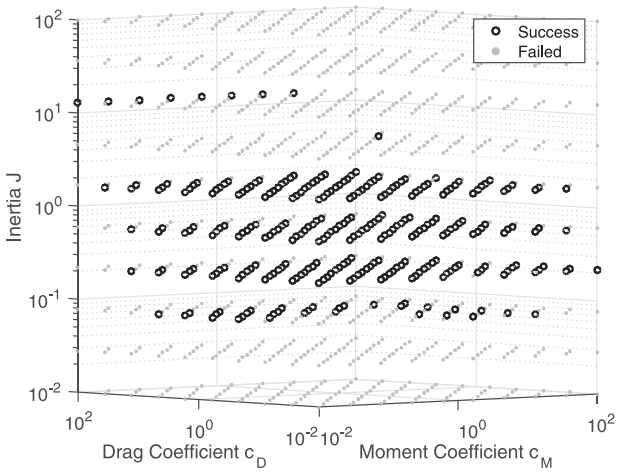

FIGURE 14 The parameters are initialized to 10 different values logarithmically scaled by 0.01 to 100 times the GT. An identification run is marked successful (black circle) when the difference between the GT and the estimated parameter is smaller than the estimated 3σ bound for all parameters. The results show that mainly for the inertia a good initial guess should be provided. This however is no issue, since a good initial guess of the inertia can be calculated by approximating the MAV with a sphere. The other two parameters converge in a wide range

to the same value even in the case of large errors on the initial parameters. This is especially important on new MAV platforms, where almost no prior knowledge is available. We therefore try to give some practical hints on how to initialize the parameters and analyze the region of attraction. We found that the initial guess can be reduced to only three unknowns, whereas the other parameters are initialized with the EKF or set to zero.

The first set of parameters is the transformation and time delay between the pose sensor and the IMU. These values are estimated with an EKF that is independent of the MAV parameters and described in more detail in Section 5.6. The location of the CoG can always be initialized with zero offset, since MAV need to be well balanced to fly. Similarly, we can assume that the inertia is comparable in all three axes, making it possible to reduce the three dimensions to one inertia parameter J for the initial guess.

Together with the drag coefficient c_D and the rotor moment coefficient c_M. this leads to the following parameter set $\theta_{\text{init}} = [c_D, c_M, J]$. We initialized the estimator with 10 different values for each axis logarithmically scaled between 0.01 to 100 times the GT value. The results can be seen in Figure 14 where a successful parameter estimate is indicated with a black circle. We define an identification run as successful in cases where the difference between the GT and the estimated parameter is smaller than the estimated 3σ bounds.

We found that the most critical parameter for a successful convergence is the inertia. Luckily, an initial guess for this parameter is easy to estimate. Either the MAV can be approximated with a sphere requiring only the mass and the diameter or an initial guess from CAD is available. The other two parameters are less critical, and the estimation is successful even in the case of a large initial error.

Together with the previous experiments, this shows that we have developed a generic identification framework for MAVs. In combination with the practical hints given, this approach should converge to the correct solution even in the case of large initial uncertainty about the parameters.

8 | CONCLUSION

In this paper, we have shown a framework for identifying the physical parameters of an MAV. By formulating the estimation problem as a ML optimization, we are not only able to estimate the mean, but also the uncertainty of the individual parameters. We carefully devised how to include dynamic systems in a ML estimation problem and applied it to MAV calibration. Furthermore, we showed how to incorporate unknown time offsets due to different clock sources. This allows us to directly include the information provided by our MAV dynamic model, the IMU, and a generic pose sensor with minimal preprocessing.

Our results show that the approach works even in the case of a large initial parameter error. Furthermore, we showed that it is possible to estimate the parameters using only on-board measurements, making it independent of any expensive or external equipment. This allows for in-field calibration in cases the platform changed given that a wind protected area is available. To help other researchers, we have also provided a number of practical hints on how to initialize the estimator and the unknown parameters. This work is an important step toward an easy to use MAV calibration toolbox, which allows for accurate simulation and enables model-based estimation and control.

ACKNOWLEDGMENTS

The authors wish to express their gratitude to Markus W. Achtelik for the many useful discussions and the great support with the platform. We also want to thank Janosch Nikolic, Igor Gilitschenski, and Hannes

Sommer for the numerous discussions and guidance on ML estimation, stochastic models, and estimation involving rotations. Finally, we want to thank Helen Oleynikova for the numerous rounds of feedback and great help with the English language.

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement no. 608849 (EuRoC) and the Swiss National Science Foundation (grant number 200021_149427/1).

REFERENCES

1. Mellinger D, Lindsey Q, Shomin M, Kumar V. Design, modeling, estimation and control for aerial grasping and manipulation. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Piscataway, NJ: IEEE; 2011: 2668–2673.
2. Bloesch M, Omari S, Hutter M, Siegwart R. Robust visual inertial odometry using a direct ekf-based approach. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Piscataway, NJ: IEEE; 2015: 298–304.
3. Bloesch M, Sommer H, Laidlow T, et al. A primer on the differential calculus of 3D orientations. arXiv preprint 1606.05285; 2016.
4. Furgale P, Rehder J, Siegwart R. Unified temporal and spatial calibration for multi-sensor systems. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Piscataway, NJ: IEEE; 2013: 1280–1286.
5. Furrer F, Burri M, Achtelik WM, Siegwart R. RotorS — A modular Gazebo MAV simulator framework. In: *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, New York: Springer International Publishing; 2016: 595–625.
6. Burri M, Nikolic J, Oleynikova H, Achtelik MW, Siegwart R. Maximum likelihood parameter identification for MAVs. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Piscataway, NJ: IEEE; 2016: 4297–4303.
7. Pounds P, Mahony R, Corke P. Modelling and control of a quadrotor robot. In *Proceedings of Australasian Conference on Robotics and Automation 2006*. Sydney, Australia: Australian Robotics and Automation Association, Inc.; 2006.
8. Derafa L, Madani T, Benallegue A. Dynamic modelling and experimental identification of four rotors helicopter parameters. In: *2006 IEEE International Conference on Industrial Technology*. Mumbai, India: IEEE; 2006: 1834–1839.
9. Hoffer NV, Coopmans C, Jensen AM, Chen Y. A survey and categorization of small low-cost unmanned aerial vehicle system identification. *J Intell Robot Syst*. 2014;74(1–2):129–145.
10. Morelli EA. Real-time parameter estimation in the frequency domain. *J Guid Control Dyn* 2000;23(5):812–818.
11. Chowdhary G, Jategaonkar R. Aerodynamic parameter estimation from flight data applying extended and unscented Kalman filter. *Aerospace Sci Technol* 2010;14(2):106–117.
12. Chaturvedi NA, Bernstein DS, Ahmed J, Bacconi F, McClamroch NH. Globally convergent adaptive tracking of angular velocity and inertia identification for a 3-DOF rigid body. *IEEE Trans Control Syst Technol*. 2006;14(5):841–853.
13. Rashid MI, Akhtar S. Adaptive control of a quadrotor with unknown model parameters. In *2012 9th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. Piscataway, NJ: IEEE; 2012: 8–14.
14. Tischler MB, Remple RK. *Aircraft and Rotorcraft System Identification*. AIAA education series. Reston, VA: AIAA; 2006.
15. Fletcher JW. A model structure for identification of linear models of the uh-60 helicopter in hover and forward flight. NASA TM: 110362, 1995.
16. Mettler B, Tischler MB, Kanade T. System identification of small-size unmanned helicopter dynamics. In *Annual Forum Proceedings- American Helicopter Society Annual Forum Proceedings*, vol. 2. Fairfax, VA: AHS; 1999: 1706–1717.
17. Jategaonkar R. *Flight Vehicle System Identification: A Time Domain Methodology*, vol. 216. Reston, VA: AIAA; 2006.
18. Jategaonkar RV, Plaetschke E. Identification of moderately nonlinear flight mechanics systems with additive process and measurement noise. *J Guid Control Dyn* 1990;13(2):277–285.
19. Weiss S, Achtelik MW, Lynen S, Chli M, Siegwart R. Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, Piscataway, NJ: IEEE; 2012: 957–964.
20. Kelly J, Sukhatme GS. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *Int. J. Robot Res.* 2011;30(1):56–79.
21. Li M, Mourikis AI. Online temporal calibration for camera–IMU systems: Theory and algorithms. *Int. J. Robot. Res.* 2014;33(7):947–964.
22. Fleps M, Mair E, Ruepp O, Suppa M, Burschka D. Optimization based IMU camera calibration. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Piscataway, NJ: IEEE; 2011: 3297–3304.
23. Nikolic J, Burri M, Gilitschenski I, Nieto J, Siegwart R. Nonparametric extrinsic and intrinsic calibration of visual-inertial sensor systems. *IEEE Sens J*. 2016;16(13):5433–5443.
24. Leutenegger S, Lynen S, Bosse M, Siegwart R, Furgale, P. Keyframe-based visual–inertial odometry using nonlinear optimization. *Int. J Robot Res* 2015;34(3):314–334.
25. Hertzberg C, Wagner R, Frese U, Schröder L. Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Inform Fusion* 2011;14(1):57–77.
26. Strasdat H, Montiel JM, Davison AJ. Visual slam: why filter? *Image Vis Comput* 2012;30(2):65–77.
27. Burri M, Bloesch M, Schindler D, Gilitschenski I, Taylor Z, Siegwart R. Generalized information filtering for MAV parameter estimation. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Piscataway, NJ: IEEE; 2016: 3124–3130.
28. Kaess M, Ranganathan A, Dellaert F. iSAM: Incremental smoothing and mapping. *IEEE Trans Robot*. 2008;24(6):1365–1378.
29. Maybeck PS. *Stochastic Models, Estimation, and Control*. New York: Academic Press; 1979.
30. Forster C, Carlone L, Dellaert F, Scaramuzza D. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In: *Robotics: Science and Systems XI;* 2015.
31. Lynen S, Achtelik MW, Weiss S, Chli M, Siegwart R. A robust and modular multi-sensor fusion approach applied to MAV navigation. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Piscataway, NJ: IEEE; 2013: 3923–3929.
32. Kabsch W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr . A*. 1976;32(5):922–923.
33. Bangura M, Melega M, Naldi R, Mahony R. Aerodynamics of rotor blades for quadrotors. arXiv preprint :1601.00733; 2016.
34. Mahony R, Kumar V, Corke P. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE RobotiAutom Mag*. 2012;19(3):20–32.

35. Omari S, Hua M-D, Ducard G, Hamel T. Nonlinear control of VTOL UAVs incorporating flapping dynamics. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Piscataway, NJ: IEEE; 2013: 2419–2425.
36. Hoffmann GM, Huang H, Waslander SL, Tomlin CJ. Precision flight control for a multi-vehicle quadrotor helicopter testbed. *Control Eng Pract* 2011;19(9):1023–1036.
37. Nikolic J, Rehder J, Burri M, et al. A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time slam. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Piscataway, NJ: IEEE; 2014: 431–437.
38. Leishman GJ. *Principles of Helicopter Aerodynamics with CD Extra*. Oxford, UK: Cambridge University Press; 2006.
39. Harris CM, Piersol, AG. *Harris' Shock and Vibration Handbook*, volume 5. New York, NY: McGraw-Hill; 2002.
40. Bähnemann R, Burri M, Galceran E, Siegwart R, Nieto J. Sampling-based motion planning for active multirotor system identification. arXiv preprint :1612.05143; 2016.

How to cite this article: Burri M, Bloesch M, Taylor Z, Siegwart R, Nieto J. A framework for maximum likelihood parameter identification applied on MAVs. *J Field Robotics*. 2018;35:5–22. https://doi.org/10.1002/rob.21729