Sampling-based Motion Planning for Active Multirotor System Identification

Rik Bähnemann, Michael Burri, Enric Galceran, Roland Siegwart, and Juan Nieto

Abstract—This paper reports on an algorithm for planning trajectories that allow a multirotor micro aerial vehicle (MAV) to quickly identify a set of unknown parameters. In many problems like self calibration or model parameter identification some states are only observable under a specific motion. These motions are often hard to find, especially for inexperienced users. Therefore, we consider system model identification in an active setting, where the vehicle autonomously decides what actions to take in order to quickly identify the model. Our algorithm approximates the belief dynamics of the system around a candidate trajectory using an extended Kalman filter (EKF). It uses sampling-based motion planning to explore the space of possible beliefs and find a maximally informative trajectory within a user-defined budget. We validate our method in simulation and on a real system showing the feasibility and repeatability of the proposed approach. Our planner creates trajectories which reduce model parameter convergence time and uncertainty by a factor of four.

I. INTRODUCTION

Multirotors are becoming increasingly popular in research, industry, and consumer electronics, with applications in aerial photography, film making, delivery, construction work, and search and rescue operations. To achieve such complex tasks autonomously, precise maneuvering of the MAV is required, which in turn requires accurate state estimation, planning, and control [1]. Those three components benefit greatly from an accurate aircraft model. During aggressive maneuvers a nonlinear model can help compensate for significant aerodynamic effects that impact the vehicle [2]. In manipulation or transportation, where the external forces change regularly, effective models can be used to estimate these disturbances [3]. Furthermore, one can use a system model to create a realistic simulation environment [4], avoiding the need to carry out expensive real-world experiments.

Estimating the parameters of such nonlinear motion models is a challenging task. A common approach is to carry out specific experiments and simulations for each parameter, for example measuring the drag coefficients in wind tunnel experiments, or approximating the moments of inertia with estimates from a CAD model. These experiments, however, are expensive, time-consuming, and require significant expertise.

A more convenient alternative consists of recording sensor data during a flight and running a maximum-likelihood (ML) batch optimization [5] or Kalman filtering on the data. These



Fig. 1. A MAV calibration routine generated by our proposed planner. The trajectory is generated by maximizing the information about the system parameters, i.e., its moments of inertia and aerodynamic coefficients. The planner creates safe trajectories that respect state and input constraints. The framework enables non-expert users to perform automated calibration.

approaches are especially useful if the robot needs to be re-calibrated often, for example when changing the hardware setup, or the payload in a package delivery situation.

As some parameters are only observable under special motions, ML methods require either a short, information-rich dataset to obtain parameter estimates with a low uncertainty, or large volumes of data which increases the computational complexity of the identification process. Designing repeatable experiments that excite all of the system's dynamics is difficult as it requires deep understanding of the model characteristics. Transferring this knowledge into feasible teleoperated robot trajectories is a further substantial challenge.

In this paper we present an automated trajectory generation framework for MAV parameter estimation based on samplingbased motion planning under uncertainty. The framework actively designs a repeatable and persistently informative experiment for calibration. The algorithm performs a tree based search over various candidate trajectory segments selecting the combination that minimizes parameter uncertainty. Our planner incorporates time, space, and control feasibility constraints. The central contributions of this work are:

- A real-time MAV motion planning framework for parameter identification experiments which requires minimum human interaction and a small number of tuning variables.
- The formulation of an EKF for parameter identification and propagation of uncertainties in rapidly-exploring random belief trees (RRBTs).
- Computational improvements to RRBTs with the inclusion of one-step propagations.

R. Bähnemann, M. Burri, R. Siegwart, and J. Nieto are with the Autonomous Systems Lab (ASL), ETH Zürich, Zürich, Switzerland. E. Galceran was also with the ASL at the time this work was conducted. {brik, burrimi, enricg, rsiegwart, jnieto}@ethz.ch This research was supported in part by the European Community's Seventh Framework Programme (grant number n.608849).

• The adaptation of an information criterion suited for robot system identification.

We organize the paper as follows. Section II presents related work. Section III states the formal problem definition. We then outline the proposed information gathering algorithm in section IV. In section V and VI we detail two main components of the algorithm: motion planning and belief propagation. Finally, we validate the results in section VIII before we close the article with concluding remarks.

II. RELATED WORK

Designing an optimal experiment for calibration requires making several design decisions: which signals to measure, where to position sensors, how to filter measurements, and how to actuate the robot. These choices should be made to maximize the observability of the modes of interest. Finding an informative trajectory crosses both the domain of system identification and motion planning. In order to classify a generated trajectory as being informative, one has to make an assessment of its quality for identification.

There exist a number of works in the literature that have presented methods for designing informative trajectories. To the authors' knowledge, the work that is most similar to ours in scope is [6]. In this work the authors optimize polynomial trajectories to calibrate an IMU-GPS MAV model. In contrast to our work they use a parameter information measure based on the observability Gramian and a continuous optimization. While their approach seems promising to overcome discretization and linearization errors which occur in tree-based approaches, they lack real-time capabilities and may be prone to local minima due to the high-dimensional non-linear space.

Another common approach to find an informative trajectory is to evaluate and optimize the statistical properties of the applied ML estimator. In [7] the authors generate trajectories for end effector calibration. In a constrained non-linear optimization they generate and refine a finite sum of harmonics trajectory to perform a batch least squares estimation. In this optimization they either minimize the regressor's condition number or maximize the Fisher information matrix. The same optimization objective is used in [8], [9] to refine an arbitrary initial input signal for a pendulum cart identification.

Instead of optimizing a subset of trajectories, one could attempt solving the information gathering problem directly in the whole configuration space. Solving this exactly is non-deterministic polynomial-time (NP)-hard [10]. Consequently, we need to find a trade-off between optimality and computational feasibility.

Recent work on sampling-based information gathering has shown to rapidly lead to near optimal solutions. Because rapidly-exploring random trees (RRTs) are biased towards unexplored areas, RRTs can quickly cover a large subset of the configuration space. This makes them real-time capable even in high dimensional spaces and due to the random nature explore different topologies to overcome local minima.

Rapidly-exploring information gathering (RIG) [11] allows solving general information gathering problems by sampling



Fig. 2. Parameter belief propagation: two different polynomial segments and the predicted moment of inertia covariance Σ_J depicted as ellipsoids. Left: the forward movement only gives information about the pitch inertia. Right: the maneuver excites roll, pitch, and yaw simultaneously and shrinks the parameter covariance in all directions. Note that the smaller the covariance matrix, the more informative the trajectory.

the configuration space. The algorithm extends previous work to submodular cost functions for which it can guarantee asymptotic optimality.

Belief graph search such as RRBT [12] or belief roadmap (BRM) [13] on the other hand use random trees to efficiently predict the EKF state covariance in a graph of motions. In [14] they use RRBT to perform motion- and uncertainty-aware path planning for MAVs.

We build upon this work to formulate a real-time capable multirotor parameter estimation framework. In contrast to [14] our work focuses on pure information gathering in a confined space and thus omits finding a goal path. Furthermore, we propose various algorithmic improvements to enhance computational times of RRBTs and decrease tuning efforts.

III. PROBLEM DEFINITION

Our goal is to find a calibration trajectory that maximizes the information about unknown MAV parameters. Equivalently, we search for egomotions which will minimize our parameter estimation's uncertainty. We formulate the problem of finding an optimal experiment design for MAV parameter estimation as the following optimization problem:

$$\mathbf{P}^{*} = \underset{\mathbf{P} \in \mathcal{P}}{\operatorname{arg\,min}} \quad u(\mathbf{P}) \tag{1}$$

s.t.
$$\mathbf{COST}(\mathbf{P}) \leq C,$$

$$\mathbf{STATES}(\mathbf{P}) \in \mathcal{X},$$

where we wish to find within all possible trajectories \mathcal{P} , a path **P** which minimizes the parameter uncertainty $u(\mathbf{P})$, such that all constraints are satisfied. The cost of the path is given by $COST(\mathbf{P})$ and is required to stay within a budget C. The robot states along the trajectory, given by $STATES(\mathbf{P})$, are constrained to remaining within the set of all feasible states \mathcal{X} .

A. Optimization Objective and Constraints

We describe the uncertainty of our unknown parameters Θ through the predicted parameter covariance matrix Σ_{Θ} of our estimator. A small covariance matrix indicates an informative path. Fig. 2 illustrates the effect of two different planned egomotions on the parameter covariance.

The utility function $u : \mathbb{R}^j \to \mathbb{R}$ quantifies the uncertainty and we optimize the trajectory in a D-optimal sense, i.e., we minimize Σ_{Θ} 's determinant. Given Σ_{Θ} 's eigenvalues λ , the D-optimal criterion is computed as

$$u(\mathbf{P}) = \exp(\frac{1}{j} \sum_{i=1}^{j} \log(\lambda_i)).$$
(2)

D-optimality is proportional to the volume of the *j*-dimensional ellipsoid spanned by the covariance matrix. In contrast to A-optimality or E-optimality, D-optimality weights each eigenvalue equally. For example a decrease of one eigenvalue by 10% decreases the total cost by 10% independent of the eigenvalue's parameter unit. This reduces tuning effort as the cost function does not require normalization.

We calculate the D-optimality in (2) in logarithmic space to avoid round off errors in case of small eigenvalues [15]. Note that we choose to penalize only parameter uncertainty rather than combinations such as the uncertainty and flight time or control effort. This decision was made as it simultaneously simplifies the planner and reduces the difficulty of tuning the system.

To ensure that the calibration routine finishes in a given time frame, we define the trajectory's cost by its total flight time t_f . Other measures such as power consumption could also be used for the cost measure as long as the function is strictly increasing over time.

$$\operatorname{COST}(\mathbf{P}) = t_f. \tag{3}$$

State constraints ensure the feasibility of the planned path. Those can arise from actuator limits, sensor, and space constrains. In particular, we place limits on absolute thrust T, angular rates $\boldsymbol{\omega}$ in base frame, and yaw acceleration $\ddot{\psi}$ in base frame. In addition we limit the robot to stay within a bounding box by constraining the position **p** in world frame.

$$0 \le T_{\min} \le T \le T_{\max}, \ T = ||\mathbf{\ddot{p}} + \mathbf{G}||,$$
$$||\boldsymbol{\omega}|| \le \omega_{\max}, \ |\mathbf{\ddot{\psi}}| \le \mathbf{\ddot{\psi}}_{\max}, \ \mathbf{p}_{\min} \le \mathbf{p} \le \mathbf{p}_{\max},$$
(4)

where $\mathbf{G} = \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T$ is the gravitational acceleration in world coordinates. Choosing conservative limits ensures that the position controller can track the trajectory closely. This is important since we use the nominal trajectory to propagate the beliefs in the planner.

Note that most controllers are tuned independently of the estimated system parameters. In case the same model is used for the controller one could think of an iterative approach where the controller is updated after each identification run.

B. Robot Model and EKF Parameter Uncertainty Prediction

We assume continuous-time nonlinear system dynamics and measurements to describe our robot state \mathbf{x} and sensor measurements \mathbf{z} :

$$\frac{d}{dt}\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w}), \qquad \mathbf{z} = \mathbf{h}(\mathbf{x}, \mathbf{v}), \qquad (5)$$

where \mathbf{u} is the control input and \mathbf{w} and \mathbf{v} are Gaussian noise accounting for modeling and measurement errors.

We augment our state vector with constant unknown parameters Θ and formulate an EKF for their estimation. During planning we sample the polynomial trajectory and calculate the nominal states, measurements, and control inputs along candidate trajectories. With these we propagate the EKF covariance to predict the probabilistic state distribution.

The parameter covariance Σ_Θ is then part of the full covariance matrix Σ

$$\boldsymbol{\Sigma} = \begin{bmatrix} \cdot & \cdot \\ \cdot & \boldsymbol{\Sigma}_{\Theta} \end{bmatrix} \tag{6}$$

and is computed at time step k, given the prior covariance $\Sigma_{k-1|k-1}$, as

$$\boldsymbol{\Sigma}_{k|k-1} = \mathbf{F}_k \boldsymbol{\Sigma}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k, \tag{7}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{\Sigma}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k, \tag{8}$$

$$\mathbf{K}_k = \mathbf{\Sigma}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}, \tag{9}$$

$$\boldsymbol{\Sigma}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \boldsymbol{\Sigma}_{k|k-1}, \qquad (10)$$

where we obtain the discrete state transition matrix \mathbf{F}_k , process noise covariance matrix \mathbf{Q}_k , observability matrix \mathbf{H}_k , and measurement noise covariance matrix \mathbf{R}_k by linearizing the system about the nominal robot trajectory. Note that we omit state prediction and updates which are unnecessary under the assumption that the MAV is able to follow the trajectory closely.

IV. INFORMATION GATHERING ALGORITHM

We approximate the solution to the minimization problem presented in (1) with a sampling-based information gathering algorithm. Our planner generates a graph of random feasible motions to explore the robot's configuration space. For every trajectory within the graph the planner computes the probability distribution over the states and parameters. Finally, the trajectory with the least uncertainty in parameters is selected. We follow the ideas of RIG [11] on sampling-based information gathering and RRBT [12] on belief space search.

In RIG the planner searches for a maximum informative path. It samples random motions and evaluates an arbitrary information criterion for each candidate path. In our case the covariance measure stated in (2) describes the information we gather about our parameter estimates. The parameter uncertainty, however, is a function of the state covariance Σ . This requires us to keep track of the state's probabilistic distribution over all sampled trajectories. RRBTs do not only explore the configuration space, but calculate the probabilistic distribution for each state by propagating an EKF, effectively searching the robot's belief space.

We adapt the original RRBT algorithm such that it does not search for a goal connection but rather, only searches the belief space and minimizes the parameter uncertainty. We also introduce the budget constraint C. The planner differentiates between *open* and *closed* trajectories. Furthermore, we do not consider deviations of the robot from the planned path, but assume that the controller can follow the trajectory closely. We also interconnect all vertices in the graph rather than only connecting neighbors as in RRBT.

Algorithm 1 shows the graph search. It consists of two main parts: (i) building a bidirectional graph of motions and (ii) searching the graph for the trajectory that minimizes parameter uncertainty.

Algorithm 1 Motion planning for MAV parameter estimation

1: $b.\Sigma := \Sigma_0; b.c := 0; b.b_p := \text{NULL}; b.v_r := v$ 2: $v.\mathbf{x} := \mathbf{x}_0; v.B_{\text{open}} := \{b\}; v.B_{\text{closed}} := \{\};$ 3: $V := \{v\}; E := \{\}; Q := \{\}$ 4: while $t_{\text{runtime}} < t_{\text{terminate}}$ do 5: % Phase 1: motion planning 6: $\mathbf{x}_{new} := SAMPLE()$ 7. $v_{\text{nearest}} := \text{NEAREST}(V, \mathbf{x}_{\text{new}})$ 8: if $\exists (e_{\text{new}} := \text{CONNECT}(\mathbf{x}_{\text{new}}, v_{\text{nearest}}, \mathbf{x}))$ then CONTINUE 9: 10: $E := E \cup e_{new}$ $V := V \cup v(\mathbf{x}_{\text{new}})$ 11: 12: $Q := Q \cup v_{\text{nearest}}.B_{\text{open}}$ $\dot{E} := \dot{E} \cup \text{CONNECT}(v_{\text{nearest}}.\mathbf{x}, \mathbf{x}_{\text{new}})$ 13: 14: for all $v \in V$ do if $\exists e_{new} := CONNECT(v.x, x_{new})$ then 15: $E := E \cup e_{\rm new}$ 16: 17: $Q := Q \cup v.B_{\mathrm{open}}$ $E := E \cup \text{Connect}(\mathbf{x}_{\text{new}}, v.\mathbf{x})$ 18: % Phase 2: belief propagation 19: while $Q \neq \emptyset$ do 20: $b := \operatorname{POP}(Q)$ 21: for all $v_{adj}(b)$ do ▷ reachable vertices 22: $b_{\text{new}} := \text{PROPAGATE}(e_{\text{adj}}, b)$ 23: if APPENDBELIEF (v_{adj}, \dot{b}_{new}) then 24: 25: $Q := Q \cup b_{\text{new}}$ if $ISOPEN(b_{new})$ then 26: $v_{\text{adj}}.B_{\text{open}} = v_{\text{adj}}.B_{\text{open}} \cup b_{\text{new}}$ 27: 28. else $v_{\text{adj}}.B_{\text{closed}} = v_{\text{adj}}.B_{\text{closed}} \cup b_{\text{new}}$ 29: 30: return $\mathbf{P} := \text{GETDOPTIMALPATH}()$

The motion graph consists of a set of vertices $v \in V$ which represent robot states **x**. These vertices are connected by bidirectional edges $e \in E$ allowing the robot to transition between two states. There are multiple paths through the graph to reach one vertex. This history of propagation is stored in belief nodes b which reside at each vertex. Each belief node describes a unique trajectory through the graph. It contains the state covariance Σ , the cost c, its parent belief node b_p , and resides at vertex v_r .

In the motion planning phase (i) the planner uniformly samples feasible states and attempts to connect them to all other existing vertices: forwards and backwards. In the belief propagation phase (ii) the planner evaluates all possible new connections. It propagates all adjacent *open* beliefs of a new vertex through every new path combinations. The planner iterates between the two phases until its runtime $t_{\rm runtime}$ exceeds the termination time $t_{\rm terminate}$.

V. MOTION PLANNING

In our motion planning, the motions are represented as 4D-polynomial segments in position, yaw, and time as in [14]. This helps to reduce the high dimensional state space of the MAV and quickly cover large sections of the configuration space with a low number of samples. With this approach a first feasible solution is found after a short time and from

then on the solution improves as the planner has more time to explore the space. Due to the differential flatness property of MAVs it is guaranteed that the MAV is able to follow them, as long as the dynamic constraints (4) are not violated.

In our case the MAV flat state σ contains position, yaw, and its derivatives [16]:

$$\boldsymbol{\sigma} = \begin{bmatrix} \mathbf{p}^T \ \dot{\mathbf{p}}^T \ \ddot{\mathbf{p}}^T \ \ddot{\mathbf{p}}^T \ \ddot{\mathbf{p}}^T \ \ddot{\mathbf{p}}^T \ \dot{\boldsymbol{\psi}}^T \ \dot{\boldsymbol{\psi}}^T \end{bmatrix}^T.$$
(11)

The algorithm starts with SAMPLE which returns a feasible position and yaw angle, from a uniform distribution, leaving the derivatives free. The first CONNECT creates a minimumsnap and minimum-angular-velocity 4D-polynomial segments towards that state. This connection locks all state derivatives. All other connections are then fully constrained up to snap and yaw acceleration. For translational P^{th} -order minimum-snap polynomials p(t) with coefficients $\mathbf{a} = [a_0, a_1, \dots, a_N]^T$ the optimization is given by:

$$p(t) = \mathbf{t} \cdot \mathbf{a}, \ \mathbf{t} = \begin{bmatrix} 1 & t & t^2 & \cdots & t^{P-1} \end{bmatrix},$$
(12)
$$\min \int_0^{t_s} \left\| \frac{d^4 p(t)}{dt^4} \right\| dt,$$

$$\text{s.t.} \frac{d^m p(0)}{dt^m} = \frac{d^m (v_{\text{start}} \cdot \mathbf{x} \cdot \mathbf{p})}{dt^m}, \ m = 0, \dots, 4,$$

$$p(t_s) = v_{\text{end}} \cdot \mathbf{x} \cdot \mathbf{p},$$

$$\frac{d^m p(t_s)}{dt^m} = \frac{d^m v_{\text{end}} \cdot \mathbf{x} \cdot \mathbf{p}}{dt^m} \text{ or free, } m = 1, \dots, 4.$$
(13)

Minimum-angular-velocity yaw segments can be derived analogously. Note that the trajectories could also be optimized with respect to other derivatives which is subject of future investigations. We solve the minimization (13) with the optimization presented in [17].

We enforce continuity up to snap and yaw acceleration at the start and goal state and thus generate 10th-order polynomial segments in position and 6th-order in yaw. This implies continuous rotor speeds at the vertices as shown in section VI-B. It allows smooth state and belief propagation as it prevents further nonlinearities due to discontinuous inputs.

Furthermore, we sample the segment time t_s for every new segment uniformly. Given the Euclidean distance d, maximum allowed absolute velocity v_{max} , and threshold $t_{s,\text{max}}$ on the maximum segment time the sample range is:

$$t_s \in \left[\frac{d}{v_{\max}}, \min(C - \min(\operatorname{Cost}(v_{\text{start}}.B_{\text{open}})), t_{s,\max})\right].$$
(14)

The lower limit is the physically achievable minimum segment time. The upper limit is either the maximum budget left at the start vertex or a fixed maximum allowed time. The budget constraint ensures that at least one belief can be propagated. The time threshold constraint reduces the search space further, i.e., we do not allow slow segments which have shown to be uninformative.

We apply the recursive algorithm presented in [18] to check a segment for feasibility as stated in (4). We extend this test for maximum yaw rates and accelerations by evaluating the roots of the appropriate polynomial. Note that these checks are



Fig. 3. The three steps of belief prediction along a candidate polynomial segment: (i) sample flat states, (ii) recover linearization states, (iii) propagate the covariance along the segment using the EKF.

independent of the estimated model parameters and neglect drag force.

VI. BELIEF PROPAGATION

Given a candidate 4D-polynomial segment and a MAV motion and measurement model, we can predict the change of the state covariance from an initial belief $\Sigma_{0|0}$ to a terminate belief $\Sigma_{N|N}$. The PROPAGATE method shown in Algorithm 1 has three steps: (i) sample the nominal flat states along the segment, (ii) recover and linearize the system about the full expected states, measurements, and inputs, and (iii) propagate the covariance using the EKF. We visualize the process in Fig. 3 and describe the key components in the following.

A. MAV Model

We formulate the EKF for a simple multirotor dynamics model as proposed in [19]. Our approach allows exchanging this model for any model that needs to be calibrated, e.g., a visual-inertial odometry model [20]. In this particular model, the dynamic couplings of the parameters make it hard to generate manual calibration routines [5].

The MAV state in this model consists of position \mathbf{p} in world frame, velocity \mathbf{v} in base frame, attitude quaternion \mathbf{q} describing the rotation from inertial to base frame, angular velocity $\boldsymbol{\omega}$ in base frame and the desired physical parameters. Aerodynamic effects are described through three coefficients for thrust c_T , drag c_D , and moment c_M . The moment of inertia matrix is $\mathbf{J} = \text{diag}([j_x \ j_y \ j_z])$. We define the input as the collection of the k rotor speeds n_i , $i \in \{1...k\}$.

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}^T & \mathbf{v}^T & \mathbf{q}^T & \boldsymbol{\omega}^T & \boldsymbol{\Theta}^T \end{bmatrix}^T, \quad (15)$$

$$\boldsymbol{\Theta} = \begin{bmatrix} c_T & c_D & c_M & j_x & j_y & j_z \end{bmatrix}^T, \quad (16)$$

$$\mathbf{u} = \mathbf{n} = [n_1 \ \dots \ n_k]^T. \tag{17}$$

The forces and moments acting on the MAV are the forces F_i and moments M_i acting on each rotor as well as the gravitational force such that the dynamics are given by

$$\dot{\mathbf{p}} = \mathbf{C}_{\mathcal{WB}}(\mathbf{q}) \cdot \mathbf{v} \tag{18}$$

$$\dot{\mathbf{v}} = \frac{1}{m} \sum_{i=1}^{k} \mathbf{F}_{i} - \boldsymbol{\omega} \times \mathbf{v} - \mathbf{C}_{\mathcal{WB}}^{T}(\mathbf{q}) \cdot \mathbf{G},$$
(19)

$$\dot{\mathbf{q}}_{IB} = \frac{1}{2} \begin{bmatrix} 0\\ \boldsymbol{\omega} \end{bmatrix} \otimes \mathbf{q},\tag{20}$$

$$\dot{\boldsymbol{\omega}} = \mathbf{J}^{-1} \left(\sum_{i=1}^{k} \left(\mathbf{M}_{i} + \mathbf{r}_{i} \times \mathbf{F}_{i} \right) - \boldsymbol{\omega} \times \mathbf{J} \boldsymbol{\omega} \right), \quad (21)$$

$$\Theta = \mathbf{0},\tag{22}$$

where C_{WB} describes the rotation matrix from base to world frame, *m* is the mass, \otimes is the quaternion product, and \mathbf{r}_i is the rotor location in the base frame with respect to the center of gravity (CoG). The rotor force \mathbf{F}_i is the sum of all rotor thrusts $\mathbf{F}_{T,i}$ which points upwards into ${}_{B}\mathbf{z}$ -direction in the base frame. The combined rotor drag and flapping force $\mathbf{F}_{D,i}$ at rotor *i* counteracts the MAV's body velocity. The rotor moment \mathbf{M}_i is the yaw moment induced by rotor speed n_i .

$$\mathbf{F}_{T,i} = c_T n_{i\,\mathcal{B}}^2 \mathbf{z},\tag{23}$$

$$\mathbf{F}_{D,i} = -c_T n_i^2 \begin{vmatrix} c_D & 0 & 0\\ 0 & c_D & 0\\ 0 & 0 & 0 \end{vmatrix} \left(\mathbf{v} + \boldsymbol{\omega} \times \mathbf{r}_i \right), \qquad (24)$$

$$\mathbf{F}_{i} = \mathbf{F}_{T,i} + \mathbf{F}_{D,i} + \mathbf{w}_{F}, \ \mathbf{w}_{F} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{F}), \qquad (25)$$

$$\mathbf{M}_{i} = -\epsilon_{i} c_{M} n_{i}^{2} \mathbf{g} \mathbf{z} + \mathbf{w}_{M}, \ \mathbf{w}_{M} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{M}), \quad (26)$$

where \mathbf{w}_F and \mathbf{w}_M are normally distributed white noise with variance \mathbf{Q}_F and \mathbf{Q}_M to account for modeling errors.

The MAV is provided with discrete position $\mathbf{z}_{p,k}$ and attitude measurements $\mathbf{z}_{q,k}$. In this work these measurements were provided by an external motion capture system. With this set of measurements the EKF is able to refine the covariances. The measurement model is given by

$$\mathbf{z}_{p,k} = \mathbf{p}_k + \mathbf{v}_p, \qquad \mathbf{v}_p \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_p), \qquad (27)$$

$$\mathbf{z}_{q,k} = \mathbf{q}_k \otimes \mathbf{v}_q,\tag{28}$$

$$\mathbf{v}_{q} = \begin{bmatrix} 1 & \frac{1}{2} \mathbf{\Phi} \end{bmatrix}^{T}, \qquad \mathbf{\Phi} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{q}), \tag{29}$$

$$\mathbf{z}_k = [\mathbf{z}_{p,k}^{\mathsf{T}} \ \mathbf{z}_{q,k}^{\mathsf{T}}]^{\mathsf{T}},\tag{30}$$

where \mathbf{v}_p and \mathbf{v}_q is normally distributed white noise with variance \mathbf{R}_p and \mathbf{R}_q accounting for small measurement errors in position or attitude.

B. Recovering the Nominal States and Inputs

In our formulation we assume that the MAV follows the planned trajectories exactly. Hence evaluating position, yaw, and its derivatives along a segment corresponds with evaluating the nominal states. Following [16] there exists a function f which maps the flat state σ into the full state x:

$$f: \boldsymbol{\sigma} \to \mathbf{x}.$$
 (31)

Additionally, one can compute the rotor speeds given the constant allocation matrix **A**, which maps rotor speeds into torque and thrust:

$$\begin{bmatrix} \mathbf{J}\boldsymbol{\alpha} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \\ m|\mathbf{a} + \mathbf{G}| \end{bmatrix} = \mathbf{A} \cdot \mathbf{n}^2, \tag{32}$$

where α is the angular acceleration in the base frame. This can also be recovered from the flat state.

C. One-Step Propagation

For each time step k = 1, ..., N the algorithm linearizes the system equations about the maximum likelihood state. We follow the approach presented in [21] and [22] for linearizing the attitude dynamics. The EKF covariance propagation, however, is computationally expensive and has to be repeated many times for the same edge during the planners propagation phase, e.g., if the motion generation closes a circle. In [13] the authors introduce a method of factoring the covariance to collapse this step by step calculation in (7 - 10) into a single linear transfer function $S_{0:N}$:

$$\begin{bmatrix} \cdot & \boldsymbol{\Sigma}_{N|N} \\ \cdot & \cdot \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \boldsymbol{\Sigma}_{0|0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \star \mathcal{S}_{0:N}, \quad (33)$$

where \star resembles the Redheffer star product for interconnecting two systems. We precompute and store $S_{0:N}$ for every new edge:

$$S_{0:N} = S_0 \star S_1 \star \dots \star S_N, \qquad S_k = S_k^C \star S_k^M.$$
 (34)

The motion update \mathcal{S}_k^C and measurement update \mathcal{S}_k^M are

$$\mathcal{S}_{k}^{C} = \begin{bmatrix} \mathbf{F}_{k} & \mathbf{Q}_{k} \\ \mathbf{0} & \mathbf{F}_{k}^{T} \end{bmatrix}, \quad \mathcal{S}_{k}^{M} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{H}_{k}^{T} \mathbf{R}_{k}^{-1} \mathbf{H}_{k} & \mathbf{I} \end{bmatrix}. \quad (35)$$

VII. PRUNING AND OPTIMALITY

Even though we are speeding up the calculation with polynomial segments and one-step propagations, the algorithm still has to evaluate every new possible path combination every time a new vertex is added to the graph. This is a combinatorial problem, leading to an increasing growth of beliefs and thus propagation time. In practice, the planner soon starts spending most of its time in the exhaustive search over all possible vertex connections.

Many of the evaluated beliefs however, are uninformative. For example if its trajectory is slow or does not excite desired modes. APPENDBELIEF prunes those beliefs, trading off optimality guarantees for computational feasibility.

We consider belief b_a better than belief b_b if it has smaller cost c, state uncertainty Σ , and parameter uncertainty Σ_{Θ} . We add the full state covariance Σ to the comparison, because a converged state can have a positive influence on a parameter update. We compare two matrices in a D-optimal way.

$$b_a < b_b \Leftrightarrow b_a.c < b_b.c \land b_a.\Sigma < b_b.\Sigma \land b_a.\Sigma_{\Theta} < b_b.\Sigma_{\Theta}$$
(36)

Fig. 4 shows that this pruning leads to a fraction of beliefs and smaller uncertainty in less time. The pruning shortens the propagation queue and allows quicker exploration of the configuration space. The drawback of this approach is that we lose optimality guarantees as shown for RIG by [11]. Due to coupled terms in the state covariance, we have a submodular utility function (2) but apply modular pruning (36).

Note that the planner was not globally optimal in the first place, because we sample only a subspace of the configuration space; however, as will be shown in our results, the algorithm quickly finds a solution which sufficiently excites all modes.

VIII. RESULTS

We implemented our planner in the open motion planning library (OMPL) [24] using C++ and evaluated it in simulations and real experiments. The algorithm is tested on an Intel Core i7-5600U CPU running at 2.60 GHz with 12 GB of RAM.



Fig. 4. Median and 95%-percentile of the number of belief nodes and the corresponding minimum uncertainty vs. the planner runtime in 100 runs. Modular pruning speeds up the convergence rate and leads to smaller uncertainties by pruning uninformative beliefs nodes. Further the sampling-based planner finds an informative trajectory in less than a minute which makes it significantly faster than existing optimization-based approaches [6]. The benchmark was created with [23].

A. Simulation Based Experiments

Simulation based experiments were first conducted to allow the extensive comparison of our method against ground truth. We use RotorS [4] to simulate an Ascending Technology (AscTec) Firefly hexacopter with the model given in Sec. VI-A. An artificial Vicon motion tracking system provides noisy position and attitude measurements with standard deviation 0.5 mm and 0.1° .

Fig. 5 shows the angular rates of a typical trajectory with corresponding EKF parameter estimation and one sigma bounds. Typically, the trajectories are as aggressive as possible given the input feasibility constraints. This maximizes the signal to noise ratio and the number of informative maneuvers given the budget constraint. It also tends to repeat information rich segments. Furthermore, the trajectory excites all directions to reduce uncertainty in all parameters. The planner demonstrates a tendency to select segments that excite roll, pitch, and yaw simultaneously to enable the observer to distinguish between the strongly coupled parameters j_z and c_M (see (21)).

In order for our approach to operate effectively, the planner needs the ability to predict the covariance of a segment. This can be seen in the lower part of Fig. 5 where the planner's predicted sigma bounds and the estimator's sigma bounds illustrate the quality of the belief prediction.

Next we validate the repeatability of our approach. We plan 100 parameter estimation experiments offline, simulate the flights, and run the proposed EKF offline to estimate the parameters. At the beginning of each experiment we sample the unknown physical parameters $\pm 50\%$ around the ground truth values to set the planner model and initialize the actual EKF estimation. The planner searches for only 30 s and generates trajectories with a flight time of 30 s.

We compare our results to random trajectory calibrations. These random trajectories are generated with the same graph based motion planning. A trajectory with a maximum number of different segments is then picked. This heuristic biases



Fig. 5. A typical simulated experiment: (top) angular rates vs. flight time, (bottom) EKF parameter estimation with predicted and actual sigma bounds vs. flight time. The planner designs an aggressive trajectory that excites all directions. It closely predicts the estimator's covariance. This allows the algorithm to generate a trajectory with low parameter uncertainty. Even a difficult to observe parameter like j_z converges within less than 7 s of flight data.



Fig. 6. Relative final estimation error (RE) vs. final sigma bounds for 100 simulated parameter estimation experiments with random initial parameter guess. The plot shows the results for random (RAND) and optimized (DOPT-COV) calibration trajectories. Our planner leads to smaller uncertainties and errors in parameter estimates. In particular, the strongly coupled parameters j_z and c_M are estimated with more confidence.

the random planner towards exciting all directions and fast maneuvers.

Fig. 6 shows the final sigma bounds and estimation errors for our optimized and random trajectories. Utilization of the proposed planner leads to smaller final uncertainty for all parameters. This is of particular importance for the coupled parameters that are inherently more challenging to accurately identify. We do not show j_y since the platform is almost symmetric. The thrust constant c_T converges already at hovering without any special calibration routine.

Finally, we show the convergence speed and repeatability with our approach in the statistical summary in Fig. 7. Tab. I quantifies the results of the statistical evaluation. We achieve more than $4\times$ faster convergence for j_x with our optimized trajectories. On average, the planner reduces the amount of data the estimator requires by 13 s or 70% compared to random trajectories. Furthermore, the median final uncertainty (2) is more than $4\times$ smaller with our proposed planner which



Fig. 7. Median and 95%-percentiles of EKF parameter estimations vs. flight time for 100 simulated parameter estimation experiments with random initial guess. The plot shows the results for random (RAND) and optimized (DOPT-COV) calibration trajectories. Our planner reduces convergence time and converges in all experiments showing repeatability of the approach.

makes our estimates more confident.

The results support our major claims: (i) the algorithm closely predicts the EKF parameter covariance, (ii) it covers a sufficient subspace of motions, (iii) it repeatably and quickly generates persistently exciting trajectories, (iv) it outperforms random trajectories, and (v) unlike random trajectories it actively selects trajectories that excite modes that are hard to observe.

B. Real Platform Experiments

In a second set of experiments, we validated our approach on a real AscTec Firefly with a Vicon motion tracking system and parameter independent nonlinear model predictive control (MPC) [25]. An example trajectory and the MAV executing this trajectory are shown in Fig. 1. Fig. 8 shows the actual parameter estimate outcome for 5 optimized trajectories starting from a fixed initial guess. Each trajectory was executed 3 times to check the repeatability of the resulting estimates. The planner excites roll and pitch such that j_x and j_y converge quickly, in less than 10 s, towards reasonable values. We can observe that the estimates are consistent, but one of the trajectories, the green one, differs in the final value for j_y .

		DI ANNIED	
	PARAMETER	DOPT-COV	RAND
	c_T	0.05 s	0.01 s
CONV TIME	c_D	2.32 s	$4.37\mathrm{s}$
	c_M	6.64 s	22.06 s
	j_x	1.62 s	6.49 s
	j_y	3.17 s	5.69 s
	j_z	6.46 s	20.01 s
DOPT	ALL	2.72×10^{-10}	1.21×10^{-9}

TABLE I

Convergence time of the median of the estimates (CONV TIME) and the median of the final D-optimal uncertainty (DOPT) from our 100 simulated estimates. We consider the median being converged when it reaches a ground truth error of less than 5% and stays constant. Calibrations on our optimized datasets require on average 70% less data and are $4\times$ more confident than calibrations on randomly generated data.



Fig. 8. 15 real system identification experiments with 5 different optimized trajectories. Each trajectory is executed 3 times to show the influence of the trajectory on the EKF parameter estimation. The values quickly converge towards reasonable values in less than 10 s. Only for the green trajectory the value differs significantly. We conclude that this trajectory most likely triggers effects that were not modeled in our EKF implementation.

An important finding from this result is that our simplifications to the implemented model might be too strong for the real system. Because the estimation bias occurs on the same trajectory repeatably, we conclude they stem from unmodeled aerodynamic effects and sensor delays. The EKF becomes overconfident on a particular motion. We still think that this trajectory is informative and we suggest either using a different estimator, e.g., a batch optimization [5] to perform the actual estimation on the sensor data or a more elaborate MAV model.

IX. CONCLUSION

We have presented a motion planning algorithm for finding maximally informative system identification experiments. The system is fully automated and therefore no expert knowledge about which motions to perform to achieve a good calibration is needed. Furthermore, our planner requires minimum tuning effort and no teleoperated flight skills. While we focus on MAV model identification, one can easily adapt the framework for any kind of robot configuration.

Performing the sampling in the MAVs flat space and calculating a one-step propagation function between two samples, the planner is able to efficiently search the robot's belief space. Unlike optimization based approaches our sampling-based approach can find a good solution after a few seconds, making it suitable for in-field calibration. Our method also does not require any initial guess for the trajectory which is otherwise required to avoid falling into a local minimum.

We showed in simulation that, on average, in only 30 s runtime our planner creates trajectories which lead to more than $4 \times$ quicker and more confident parameter estimates than random trajectories. Real experiments confirm the feasibility of our approach and emphasize the importance of a good model and calibration routine for MAV parameter estimation.

References

 V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles," in *The International Journal of Robotics Research (IJR)*, vol. 31, 2012.

- [2] M. Kamel, K. Alexis, M. Achtelik, and R. Siegwart, "Fast nonlinear model predictive control for multicopter attitude tracking on SO(3)," in *IEEE Conference on Control Applications (CCA)*, 2015.
- [3] M. Burri, M. Datwiler, M. W. Achtelik, and R. Siegwart, "Robust state estimation for Micro Aerial Vehicles based on system dynamics," in *IEEE Conference on Robotics and Automation (ICRA)*, 2015.
- [4] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *Robot Operating System (ROS): The Complete Reference (Volume 1)*. Springer International Publishing, 2016, ch. RotorS–A Modular Gazebo MAV Simulator Framework.
- [5] M. Burri, J. Nikolic, H. Oleynikova, M. Achtelik W., and S. Roland, "Maximum likelihood parameter identification for mavs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [6] K. Hausman, J. Preiss, G. S. Sukhatme, and S. Weiss, "Observability-Aware Trajectory Optimization for Self-Calibration with Application to UAVs," 2016, arXiv:1604.07905.
- [7] J. Swevers, C. Ganseman, D. B. Tükel, J. De Schutter, and H. Van Brussel, "Optimal robot excitation and identification," in *IEEE Transactions* on *Robotics and Automation*, vol. 13, no. 5, 1997.
- [8] A. Wilson, J. Schultz, and T. Murphey, "Trajectory Synthesis for Fisher Information Maximization," in *IEEE Transactions on Robotics (T-RO)*, vol. 30, no. 6, 2014.
- [9] A. D. Wilson, J. A. Schultz, and T. D. Murphey, "Trajectory Optimization for Well-Conditioned Parameter Estimation," in *IEEE Transactions* on Automation Science and Engineering (T-ASE), vol. 12, no. 1, 2015.
- [10] C.-W. Ko, J. Lee, and M. Queyranne, "An exact algorithm for maximum entropy sampling," in *INFORMS Operations Research*, vol. 43, no. 4, 1995.
- [11] G. A. Hollinger and G. S. Sukhatme, "Sampling-based Motion Planning for Robotic Information Gathering," in *The International Journal of Robotics Research (IJR)*, vol. 33, no. 9, 2014.
- [12] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *IEEE Conference on Robotics and Automation (ICRA)*, 2011.
- [13] S. Prentice and N. Roy, "The Belief Roadmap : Efficient Planning in Belief Space by Factoring the Covariance," in *The International Journal of Robotics Research (IJR)*, vol. 28, 2009.
- [14] M. W. Achtelik, S. Lynen, S. Weiss, M. Chli, and R. Siegwart, "Motionand Uncertainty-aware Path Planning for Micro Aerial Vehicles," in *Journal of Field Robotics (JFR)*, vol. 31, no. 4, 2014.
- [15] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active SLAM," in *IEEE Conference on Robotics* and Automation (ICRA), 2012.
- [16] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE Conference on Robotics and Automation (ICRA)*, 2011.
- [17] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *International Symposium on Robotics Research (ISRR)*, 2013.
- [18] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," in *IEEE Transactions on Robotics (T-RO)*, vol. 31, no. 6, 2015.
- [19] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," in *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, 2012.
- [20] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Realtime onboard visual-inertial state estimation and self-calibration of mavs in unknown environments," in *IEEE Conference on Robotics and Automation (ICRA)*, 2012.
- [21] S. M. Weiss, "Vision based navigation for micro helicopters," Ph.D. dissertation, ETH Zürich, 2012.
- [22] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D Attitude Estimation," in University of Minnesota, Department of Computing Science and Engineering, Tech. Rep., no. 2005-002, 2005.
- [23] M. Moll, I. A. Sucan, and L. E. Kavraki, "Benchmarking Motion Planning Algorithms: An Extensible Infrastructure for Analysis and Visualization," in *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, 2015.
- [24] I. A. Sucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," in *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, 2012.
- [25] M. Kamel, M. Burri, and R. Siegwart, "Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles," 2016, arXiv:1611.09240.