Backflipping With Miniature Quadcopters by Gaussian-Process-Based Control and Planning

Péter Antal, Tamás Péni, *Member, IEEE*, and Roland Tóth, *Senior Member, IEEE*

***Abstract*— This article proposes two control methods for performing a backflip maneuver with miniature quadcopters. First, an existing feedforward control approach is improved by finding the optimal sequence of motion primitives via Bayesian optimization, using a surrogate Gaussian process (GP) model. To evaluate the cost function, the flip maneuver is performed repeatedly in a simulation environment. The second method is based on closed-loop control and it consists of two main steps: first, a novel robust, adaptive controller is designed to provide reliable reference tracking even in case of model uncertainties. The controller is constructed by augmenting the nominal model of the drone with a GP that is trained using measurement data. Second, an efficient trajectory planning algorithm is proposed, which designs feasible trajectories for the flip maneuver using only quadratic programming. The two approaches are analyzed in simulations and in real experiments using Bitcraze Crazyflie 2.1 quadcopters.**

***Index Terms*— Aerial robotics, Gaussian process (GP), nonlinear control, robust control, trajectory planning.**

I. Introduction

With the widespread use of quadcopters, increasing expectations toward these systems, such as fast autonomous navigation in a cluttered environment, rapidly changing wind conditions in built environments, and security and surveillance tasks, require to perform complex, fast maneuvers that push the drones to their physical limits [1]. In these cases, classical flight controllers designed for a linearized dynamical model of the vehicle are no longer sufficient, and more advanced control methods capable to handle the entire operating domain are needed [2]. These algorithms can be developed based on nonlinear control techniques or machine learning approaches.

Execution of a backflip illustrates well such complex maneuvers because it requires careful handling of the full complex nonlinear behavior of the drone, and it is typically a challenging task even for a human pilot. The complexity and speed of the maneuver are characterized by the fact that it takes less than a second to complete during which the vehicle is able to make a full turn around one of the horizontal axes.

Because of its benchmark characteristics, various control strategies have already been proposed to perform the flip maneuver. In [3], energy-based control is applied to overcome the uncontrollability of the quadcopter at singular configurations when following a circular or clothoidal reference trajectory. In [4], a Lyapunov-stability based controller synthesis is used to execute multiflip maneuvers with quadcopters. Machine learning approaches are used in many cases, for example, to imitate the maneuver performed by an expert drone pilot with apprenticeship learning [5], or train a deep neural network sensorimotor controller for executing acrobatic maneuvers [6].

A simple learning strategy for adaptive feedforward control is proposed in [7], based on the optimization of a parametric motion primitive sequence. As backflipping pushes the actuators of the quadcopter to their physical limits, the application of near-maximal and minimal control inputs is required. This approach builds on the theory of bang-bang control and first-principle motion primitive design to perform and optimize the flip maneuver. The method is easy to implement and it is well-suited for generating a feasible motion sequence; however, many trials on the real robot are necessary to optimize the parameters of the motion and the resulting control law is sensitive to parameter uncertainties and external disturbances. These effects have greater influence on the behavior of miniature quadcopters compared with medium-sized and large drones, and hence, the control robustness is even more important.

The robust adaptive control method we propose[1] in this article is based on geometric control, which is a nonlinear approach for attitude feedback control of rigid bodies in 3-D space. In [2], it is theoretically proven that geometric control is able to stabilize the orientation of a quadcopter in the whole operating domain based on differential geometric considerations and Lyapunov stability. In [9], this geometric control is augmented with robust terms to guarantee uniformly ultimately bounded tracking errors in the presence of uncertainties in the quadcopter dynamics. However, the control design requires

Manuscript received 15 February 2023; accepted 26 June 2023. Date of publication 4 August 2023; date of current version 29 December 2023. This work was supported in part by the European Union within the Framework of the National Laboratory for Autonomous Systems under Grant RRF-2.3.1-21-2022-00002 and in part by the Eötvös Loránd Research Network under Grant SA-77/2021. Recommended by Associate Editor M. Ariola. *(Corresponding author: Péter Antal.)*

Péter Antal and Tamás Péni are with the Systems and Control Laboratory, Institute for Computer Science and Control, 1111 Budapest, Hungary (e-mail: antalpeter@sztaki.hu; peni@sztaki.hu).

Roland Tóth is with the Systems and Control Laboratory, Institute for Computer Science and Control, 1111 Budapest, Hungary, also with the Vehicle Industry Research Center, Széchenyi István University, 9026 Győr, Hungary, and also with the Control Systems Group, Eindhoven University of Technology, 5612 Eindhoven, The Netherlands (e-mail: tothroland@sztaki.hu).

Digital Object Identifier 10.1109/TCST.2023.3297744

[1] In the conference paper [8], preliminary version of the algorithms presented in this article has been discussed. Compared with [8], the main contributions of this article include the introduction of the Bayesian-optimization-based feedforward method, the Gaussian process (GP)-based model augmentation, and the derivation and proof of the robustness properties of the feedback control algorithm.

a priori knowledge of the magnitude of external disturbances which can be challenging to foresee in real-world situations. An adaptive augmentation of geometric control is proposed in [10], where the adaptive terms compensate the effects of uncertainties in the quadrotor dynamics, while the stability of the closed-loop system is proven mathematically. Artificial neural networks (ANNs) are used in [11] to develop an adaptive geometric control law that renders the quadcopter able to perform complex maneuvers in wind fields. Although both the adaptive algorithms can be implemented efficiently, they do not provide an estimation of the uncertainty of the adaptive terms. Furthermore, both the methods lack a systematic way to determine the parametric structure of the adaptive terms, which can be challenging without expert knowledge of the unmodeled dynamics and external disturbances.

To overcome these challenges, the main contributions of our present work are as follows.

- C1 We improve the convergence and the training time of the feedforward control design of [7] by applying Bayesian optimization with a GP surrogate function, making it possible to effectively apply the method in real experiments.
- C2 By GP-based augmentation of a nominal quadcopter model, we achieve adaption to unknown model dynamics and external disturbances together with quantification of the remaining model uncertainty. A robust geometric control scheme is designed that exploits the GP model for improved robust performance compared with previous methods and has convergence guarantees.
- C3 To execute the flip maneuver by the robust geometric approach, we propose an optimization-based trajectory planning method. The algorithm is based on quadratic programming and it is computationally efficient.
- C4 We compare the proposed methods in simulation and experimentally in performing the backflip maneuver.

This article is structured as follows. First, a brief introduction to GP regression is given in Section II, while in Section III, the dynamic motion model of quadcopters is presented. Then, the Bayesian-optimization-based improved feedforward control strategy is described in Section IV, corresponding to C1. The GP-based robust geometric reference tracking control scheme and the quadratic-programming-based trajectory planning are presented in Section V, corresponding to C2 and C3. Sections VI and VII give a detailed comparison of the two control approaches: first via simulations and then in real-world experiments, providing C4. Finally in Section VIII, conclusions on the proposed approaches are drawn. A video presentation of our results is available at https://youtu.be/Ed9jYlZr95c.

II. GP Regression

GPs are universal function approximators [12]. Due to their flexibility, wide representation capability, and ability to express uncertainty of the approximation, GPs have become popular in robotics and control engineering [13], [14], [15], [16]. Other supervised learning approaches, such as ANNs, are frequently applied to compensate unmodeled dynamic effects [11], [17]; however, the uncertainty quantification of ANNs is complex and unreliable. Basic statistical methods (e.g., Gamma tests) are not useful for robust control, parameter uncertainty with ellipsoidal regions is overly conservative, while both MCMC and dropout methods together with Bayesian ANNs are computationally overwhelming [18]. The main advantage of GPs over ANN-based methods is that they provide co-estimation of the nominal functional relationship together with its uncertainty in a computationally efficient manner as shown in many applications [12]. This makes them especially attractive for developing adaptive robust control solutions. In terms of real-time implementation, the evaluation of baseline GP requires the entire training dataset that can be computationally demanding; however, there are several methods that solve this problem efficiently [19].

GP regression is used for estimating an unknown, possibly nonlinear relationship $f_0 : \mathbb{R}^n \to \mathbb{R}$ between the input $x \in \mathbb{R}^n$ and noisy output observations $y \in \mathbb{R}$ of the form

$$y = f_0(x) + \epsilon \tag{1}$$

where ϵ is an independent noise process with $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. In fact, y and ϵ are random variables, but for the sake of simplicity, we will not use a different notation for their sample realization. Consider that a set of observations $\mathcal{D}_N = \{x_i, y_i\}_{i=1}^N$ are available from (1). The core idea of GP-based estimation of f_0 is to consider that candidate estimates f belong to a GP, seen as a prior distribution. Then, using $\mathcal{D}_N$ and this prior, a predictive GP distribution of f is computed that provides estimate of f_0 in terms of its mean and describes uncertainty of this estimate by its variance.

In terms of definition, a GP $\mathcal{GP} : \mathbb{R}^n \to \mathbb{R}$ assigns to every point $x \in \mathbb{R}^n$ a random variable $\mathcal{GP}(x) \in \mathbb{R}$ such that for any finite set $x_1, \ldots, x_N$, the joint probability distribution of $\mathcal{GP}(x_1), \ldots, \mathcal{GP}(x_N)$ is Gaussian. GPs are fully determined by their mean m and covariance functions κ, hence if $f \sim \mathcal{GP}(m, \kappa)$

$$m(x) = \mathbb{E}\{f(x)\}$$
$$\kappa(x, \tilde{x}) = \mathbb{E}\{(f(x) - m(x))(f(\tilde{x}) - m(\tilde{x}))\}$$

then the joint Gaussian probability of $\mathcal{GP}(x_1), \ldots, \mathcal{GP}(x_N)$ is $\mathcal{N}(M_x, K_{xx})$ with $M_x = [\, m(x_1), \ldots, m(x_N) \,]^\top$ and $[K_{xx}]_{i,j} = \kappa(x_i, x_j)$, $i, j \in \{1, \ldots, N\}$. Both m and κ, where the latter is also called a *kernel* function, are often parameterized in terms of *hyper parameters* $\theta \in \mathbb{R}^{n_\theta}$. In fact, taking $f \sim \mathcal{GP}(m, \kappa)$ as the prior distribution in the estimation process defines the prior knowledge about f_0 in terms of the mean function m, while the choice of κ determines the function space in which an estimate of the function is searched for. Parameterization of m and κ in terms of θ allows to adjust the prior, i.e., these choices to the estimation problem of f_0 using $\mathcal{D}_N$. For the estimation of a smooth f_0, a *squared exponential* (SE) kernel for κ is a common choice. The SE kernel is characterized by

$$\kappa_{\mathrm{SE}}(x, \tilde{x}) = \sigma_{\mathrm{f}}^2 \exp\left(-\frac{1}{2}(x - \tilde{x})^\top \Lambda^{-1}(x - \tilde{x})\right) \tag{2}$$

where hyperparameters are the scaling σ_{f}, used for numerical conditioning, and the symmetric matrix Λ, which determines the smoothness of the candidate function class along each x_i.

Based on the given $\mathcal{D}_N$ and the prior $f \sim \mathcal{GP}(m, \kappa)$

$$p(Y|X,\theta) = \mathcal{N}(M_x,\, K_{xx} + \sigma_\epsilon^2 I) \tag{3}$$

describes the probability density function of the outputs $Y = [\ y_1, \ldots, y_N\ ]^\top$ seen as random variables conditioned on the observed inputs $X = [\ x_1, \ldots, x_N\ ]^\top$ and hyperparameter values θ. To predict the value of the unknown function f_0 at a test point x_*, the following joint distribution

$$\begin{bmatrix} Y \\ f(x_*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} M_x \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K_{xx} + \sigma_\epsilon^2 I & K_x(x_*) \\ K_x^\top(x_*) & \kappa(x_*, x_*) \end{bmatrix}\right)$$

with $[K_x(x_*)]_i = \kappa(x_i, x_*)$ holds based on the previous considerations. Hence, the predictive distribution for $f(x_*)$, based on the observed samples $\{y_i\}_{i=1}^N$ in $\mathcal{D}_N$, is the posteriori $p(f(x_*)|\mathcal{D}_N, x_*) = \mathcal{N}(\mu(x_*), \sigma(x_*))$ characterized by

$$\mu(x_*) = m(x_*) + K_x^\top(x_*)\big(K_{xx} + \sigma_\epsilon^2 I_N\big)^{-1}(Y - M_x) \tag{4a}$$

$$\sigma(x_*) = k(x_*, x_*) - K_x^\top(x_*)\big(K_{xx} + \sigma_\epsilon^2 I_N\big)^{-1} K_x(x_*). \tag{4b}$$

The mean (4a) gives an approximation of $f_0(x_*)$ while the variance (4b) gives measure of the uncertainty of this approximation. Computation of (4) requires only elementary matrix operations, and therefore, it is computationally efficient.

To tune the hyperparameters θ and σ_ϵ^2 associated with the prior, a common method is to maximize the likelihood, i.e., probability, of the observations of Y for (3) marginalized with respect to θ and σ_ϵ

$$\begin{bmatrix} \theta^* \\ \sigma_\epsilon^* \end{bmatrix} = \arg\max_{\theta, \sigma_\epsilon}\ \log(p(Y|X,\theta,\sigma_\epsilon)) \tag{5}$$

where without loss of generality, the log of the pdf is taken to simplify the optimization problem and

$$\begin{aligned} &\log(p(Y|X,\theta,\sigma_\epsilon)) \\ &\quad = -\frac{1}{2}\big(Y^\top (K_{xx}^{-1} + \sigma_\epsilon^2 I_N) Y \\ &\qquad + \log\det(K_{xx}^{-1} + \sigma_\epsilon^2 I_N) + N\log(2\pi)\big). \end{aligned} \tag{6}$$

For alternative methods, see [12]. Here, we considered scalar-valued GPs; however, GP regression can be applied under multidimensional outputs by estimating a predictive distribution of each output dimension independently.

III. Quadcopter Dynamics

In this section, we introduce the basic principles of quadcopter modeling based on [20] and develop a nominal quadcopter model.

As the first step, three frames are introduced: the inertial frame $\mathcal{F}^{\mathrm{i}}$ in the *north–east–down* (NED) coordinates, the vehicle frame $\mathcal{F}^{\mathrm{v}}$ fixed to the vehicle, but aligned with $\mathcal{F}^{\mathrm{i}}$, and the body frame $\mathcal{F}^{\mathrm{b}}$ aligned with the body of the vehicle. The transformation from $\mathcal{F}^{\mathrm{i}}$ to $\mathcal{F}^{\mathrm{v}}$ is only a translation, while $\mathcal{F}^{\mathrm{v}}$ and $\mathcal{F}^{\mathrm{b}}$ are connected by rotation only [21]. In Fig. 1, the three frames are displayed with the Euler angles characterizing the pose in the body frame (roll: ϕ, pitch: θ, yaw: ψ) together with the direction of the rotor thrusts and angular velocities. Based on these frames, the dynamic model is formulated as

$$m\ddot{r} = mge_3 - FR_{\mathrm{b}}^{\mathrm{v}} e_3 \tag{7a}$$

$$\dot{R}_{\mathrm{b}}^{\mathrm{v}} = R_{\mathrm{b}}^{\mathrm{v}} \hat{\omega}^{\mathrm{b}} \tag{7b}$$

$$J\dot{\omega}^{\mathrm{b}} = \tau - \omega^{\mathrm{b}} \times J^{\mathrm{b}}\omega^{\mathrm{b}} \tag{7c}$$

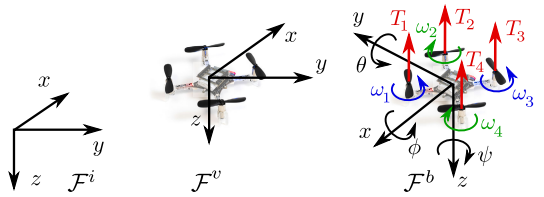


Fig. 1. Inertial $\mathcal{F}^{\mathrm{i}}$, vehicle $\mathcal{F}^{\mathrm{v}}$, and body $\mathcal{F}^{\mathrm{b}}$ frames describing the geometric relationships of the vehicle and the environment. Thrusts and angular velocities of the rotors are also illustrated.

where $r = [\ x\ y\ z\ ]^\top$ is the position of the quadcopter in the inertial frame $\mathcal{F}^{\mathrm{i}}$, e_3 is the unit vector of the z-axis in $\mathcal{F}^{\mathrm{b}}$, m is the mass of the drone, F is the collective thrust of the propellers, and g is the gravitational acceleration. $R_{\mathrm{b}}^{\mathrm{v}}(t) \in \mathrm{SO}(3)$ is the rotation matrix from $\mathcal{F}^{\mathrm{b}}$ to $\mathcal{F}^{\mathrm{v}}$, where SO(3) denotes the 3-D special orthogonal group, also called the *rotation group*. Furthermore, ω^{b} is the angular velocity of the vehicle in the body frame, J^{b} is the inertia matrix of the body of the vehicle, and $\tau = [\ \tau_x\ \tau_y\ \tau_z\ ]^\top$ is the vector of torques produced by the propellers. The notation $\hat{\cdot}$ stands for the projection: $\mathbb{R}^3 \to \mathrm{SO}(3)$ ensuring that $\hat{x}y = x \times y$ for all $x, y \in \mathbb{R}^3$ where $\times$ corresponds to the vector product. To simplify the notation, the coordinate frames are not indicated in the sequel, i.e., $R = R_{\mathrm{b}}^{\mathrm{v}}$, $J = J^{\mathrm{b}}$, $\omega = \omega^{\mathrm{b}}$.

The dynamic model has four inputs, the collective thrust F, and the torques around the three axes of the body frame τ. Assuming that the quadcopter configuration is symmetric and the torque generated by each propeller is proportional to the rotor thrust T_i, these inputs can be calculated as follows:

$$\begin{bmatrix} F \\ \tau \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -l & -l & l & l \\ l & -l & -l & l \\ \frac{b}{c} & -\frac{b}{c} & \frac{b}{c} & -\frac{b}{c} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{8}$$

where l is the distance of two motors along the x-axis, b is the drag constant, and c is the thrust constant. Furthermore, the thrust generated by each motor is considered to be proportional to the square of the corresponding angular velocity: $T_i = c\omega_i^2$ for $i \in \{1, 2, 3, 4\}$. In terms of input constraints, the individual rotor thrusts are in the range $0 \le T_i \le T_{\max}$, where $T_{\max}$ depends on the specific quadcopter design.

IV. Feedforward Control by Bayesian Optimization

A. Overview

The flip maneuver can be executed as a 360° rotation around the y-axis of the body frame of the quadcopter, displayed in Fig. 1. If the corresponding ideal actuation sequence of the individual motors in terms of T_i is computed based on the nominal quadcopter model to execute this maneuver, then the actuation sequence can be implemented on the real quadcopter in terms of feedforward control. However, computation of such an actuation profile is difficult, due to: 1) the complexity of the involved optimization problem to find a feasible motion trajectory under the given actuation constraints and 2) due to unmodeled aerodynamical effects that can significantly influence the system response.

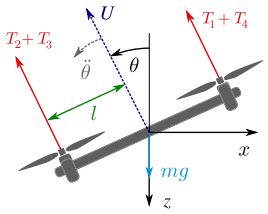


Fig. 2. Two-dimensional vehicle frame, the pitch orientation and the forces acting on the quadcopter during an ideal flip.

The feedforward control proposed in [7] solves the problem above by tuning a fixed sequence of parameterized motion primitives via experiments. However, the approximation of the Jacobian matrix in the optimization loop requires many trials on the real drone and careful selection of the measurement data is needed to ensure the numerical convergence. We have modified the original algorithm at several points to improve its performance and adapt it to our specific design configuration. First, we tune the parameters of the motion primitives in simulation using a high-fidelity nonlinear model of the drone. Second, the optimization is solved by a Bayesian optimization method, using GP surrogate function. The main advantages of the proposed method are that Bayesian optimization is numerically better conditioned and requires significantly less function evaluations than the Jacobi approximation as the evaluation points are systematically selected. This makes the proposed method computationally more favorable.

Unlike in [7], we perform the backflip in a "×" configuration as two rotors can produce a larger torque than one. The desired trajectory of the flip motion is within the xz plane of the body frame (illustrated in Fig. 2), and therefore, the equations of motion (7) can be simplified using that the translation along the y-axis and the rotation around the x- and z-axes, i.e., the roll ϕ and the yaw ψ, are fixed to zero.

The simplified equations of motion are as follows:

$$m\ddot{x} = -(T_1 + T_2 + T_3 + T_4)\sin\theta \tag{9a}$$

$$m\ddot{z} = -(T_1 + T_2 + T_3 + T_4)\cos\theta + mg \tag{9b}$$

$$J_{yy}\ddot{\theta} = l(T_1 + T_4 - T_2 - T_3). \tag{9c}$$

The directions of the thrusts and pitch are illustrated in Fig. 2.

B. Parameterized Primitives of the Maneuver

Similar to the control strategy laid down in [7], the backflip maneuver is divided into five main phases (motion primitives) that are illustrated in Fig. 3 and defined as follows.

1) *Accelerate:* Gain elevation and kinetic energy with near-maximal collective acceleration, while rotating slowly to the negative direction of the pitch angle θ.
2) *Start Rotation:* Increase the angular velocity by applying maximal differential thrust.
3) *Coast:* With low and uniform thrusts, hold the angular velocity, and wait for the drone to rotate.
4) *Stop Rotation:* Use maximal differential thrust to decrease the angular velocity and stop the rotation.
5) *Recover:* Apply near-maximal collective thrust to compensate gravity, and try to get back to hover mode.

Each of the five phases has three parameters, the collective acceleration U_i, duration t_i, and angular acceleration $\ddot{\theta}_i$, resulting in 15 parameters altogether. However, based on [7], the number of parameters can be reduced by applying bang-bang-type control on a restricted control envelope. This means that the control actions U_i and $\ddot{\theta}_i$ are either zero or near-maximal during all the phases. As a result, only the following five independent parameters remain: $\eta = \begin{bmatrix} U_1 & t_1 & t_3 & U_5 & t_5 \end{bmatrix}^\top \in \mathbb{R}^5_+$.

These parameters are tuned to minimize the norm of the final state error $e \in \mathbb{R}^5$, which is obtained by applying the actuation profile in Fig. 3 in open loop and taking the difference between the final and initial states. Formally, the optimization problem can be written as follows:

$$\begin{aligned} \min_{\eta\in\mathbb{R}^5_+} \quad & \|e(\eta)\|_2 \\ \text{s.t.} \quad & e(\eta) = [\ x(t_\text{f})\ z(t_\text{f})\ \dot{x}(t_\text{f})\ \dot{z}(t_\text{f})\ \theta(t_\text{f})\ ]^\top \\ & x(t_0) = z(t_0) = \dot{x}(t_0) = \dot{z}(t_0) = 0 \\ & U_\text{min} \le U_i \le U_\text{max}, \qquad i \in \{1, 5\} \\ & t_\text{min} \le t_j \le t_\text{max}, \qquad j \in \{1, 3, 5\} \end{aligned} \tag{10}$$

where t_0 and t_f are the initial and final time instants of the maneuver, respectively, and the bounds U_min, U_max, t_min, t_max are determined from the physical limitations of the drone.

C. Bayesian Parameter Optimization

In [7], the numerical optimization of η is based on iterative optimization, using an approximate Jacobian matrix of the final state error with respect to the parameter vector. However, the numerical gradient approximation has vast computational cost, because the whole maneuver needs to be simulated in every approximation step and it suffers from convergence problems. Here, as our first contribution, we apply a Bayesian optimization approach to find the global optimum of (10). This approach does not require the calculation of derivatives and it is suitable for global optimization of cost functions that are expensive to evaluate; see [22], [23]. To apply this approach, the optimization problem (10) is written as

$$\max_{x\in\mathcal{X}} f(x) \tag{11}$$

where $\eta = x \in \mathbb{R}^n$ is the $n = 5$ dimensional vector of optimization variables, $\mathcal{X}$ is the feasible parameter set (bounded interval of the search space for the parameters η), and f is the objective function, i.e., $f(x) = -\|e(\eta = x)\|_2$. The core concept of Bayesian optimization is to evaluate the unknown objective function at limited number of points, giving $\mathcal{D}_N = \{y_i = f(x_i), x_i\}_{i=1}^N$, fit a surrogate model based on GP regression on the data, and optimize this surrogate model of the original objective function [12]. The optimization is performed iteratively, where in each step the next evaluation point is determined by minimizing a so-called *acquisition function*, the objective function is evaluated at this point, and the surrogate model is updated. The optimization stops if the minimum is reached with high confidence or the iteration reaches a certain number of evaluations.

The acquisition function blends the approximated objective (the mean of the GP) and the approximation uncertainty (the variance of the GP) in a scalar-valued function that can be optimized by standard gradient-based procedure. A common

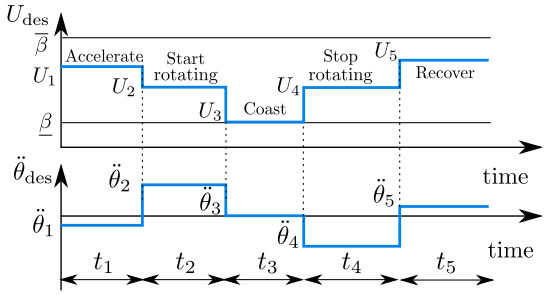


Fig. 3. Flip motion described in terms of five parameterized motion primitives.

acquisition function is *expected improvement*, defined as follows. After f is evaluated in N points, giving the observation dataset $\mathcal{D}_N$, a GP predictive distribution $\hat{f}_N \sim \mathcal{GP}(\mu_N, \sigma_N)$ is obtained with respect to $\mathcal{D}_N$. Let $\bar{f}_N^+$ be the value of the best sample so far and $x_N^+ = \arg\max_{x \in X_N} \mu^{(N)}(x)$ be the location of that sample based on $X_N = \{x_i\}_{i=1}^N$, i.e., $\bar{f}_N^+ = \mu^{(N)}(x_N^+)$. The next test point x_{N+1} is chosen such that the expected improvement predicted by the GP model is the best with respect to x_N^+

$$\mathrm{EI}_N(x) := \mathbb{E}\{\lfloor \hat{f}_N(x) - \bar{f}_N^+ \rfloor\} \tag{12}$$

where $\lfloor y \rfloor = \max(y, 0)$. The right-hand side of (12) has an analytical form and the next test point is obtained via $x_{N+1} = \arg\max_{x \in \mathcal{X}} \mathrm{EI}_N(x)$, the point with highest expected improvement. In the literature, there are other common acquisition functions, e.g., upper confidence bound or knowledge gradient [22]. Steps of the Bayesian optimization using a GP surrogate model are summarized in Algorithm 1, based on [22].

With the mathematical model of the quadcopter and a suitable optimization algorithm, it is possible to simulate the maneuver with different parameter sets, optimize the motion, and implement it on the vehicle. For the implementation of the flip maneuver, a stabilizing feedback controller is also required to balance the quadcopter at the beginning and after the end of the maneuver, for which we use the geometric control method introduced in Section V.

Algorithm 1 High-Level Steps of Bayesian Optimization

f is evaluted at $N > 0$ initial points, providing $\mathcal{D}_N$
Set a GP prior on f in terms of $\hat{f} \sim \mathcal{GP}(m, \kappa)$
while $N \leq N_{\max}$ **do**
  Determine $\hat{f}_N \sim \mathcal{GP}(\mu_N, \sigma_N)$ via (4) and (5) w.r.t. $\mathcal{D}_N$
  Let $x_{N+1} = \arg\max_{x \in \mathcal{X}} \mathrm{EI}_N(x)$,
  Observe $y_{N+1} = f(x_{N+1})$
  $\mathcal{D}_{N+1} = \mathcal{D}_N \cup \{y_{N+1}, x_{N+1}\}$
  $N \leftarrow N + 1$
end while
return $x_* = \arg\max_{x \in \mathcal{X}} \mu^{(N)}(x)$

V. Geometric Tracking Control With Trajectory Planning

The second considered approach for quadcopter backflipping is based on closed-loop control. For this purpose, as our second contribution, we propose a novel robust adaptive controller to provide reliable reference tracking even in the case of modeling uncertainties and external disturbances. The proposed method is an extension of the geometric control law [9] applicable for trajectory tracking of aggressive maneuvers. Furthermore, we also introduce a novel optimization-based trajectory planning method for this geometric approach, which is essential for finding an efficient motion path for backflipping.

A. Robust Adaptive Geometric Control by GPs

The proposed robust nonlinear geometric tracking control is based on the extension of the control law introduced in [9]. The control method is able to track a reference position $r_\mathrm{d}(t) = [\ x_\mathrm{d}(t)\ y_\mathrm{d}(t)\ z_\mathrm{d}(t)\ ]^\top$ and a reference attitude $R_\mathrm{d}(t) \in \mathrm{SO}(3)$, represented by rotation matrices. To synthesize the control law, we use (7), describing the quadcopter dynamics, and augment it by an additive state-dependent disturbance

$$m\ddot{r} = mge_3 - FRe_3 + \Delta_\mathrm{r}(\mathcal{X}) \tag{13a}$$

$$J\dot{\omega} = \tau - \omega \times J\omega + \Delta_\mathrm{R}(\mathcal{X}) \tag{13b}$$

where $\Delta_\mathrm{r}(\mathcal{X}), \Delta_\mathrm{R}(\mathcal{X}) \in \mathbb{R}^3$ comprise the model errors and uncertainties associated with the quadcopter dynamics, which depend on the state vector $\mathcal{X} = [\ r^\top\ \dot{r}^\top\ q^\top\ \omega^\top\ ]^\top$. The attitude quaternion q is computed directly from R.

For controlling the flight dynamics in (13), we propose to use the control law in [9] augmented by the adaptive state-dependent terms $\eta_\mathrm{r}, \eta_\mathrm{R}$ to cope with Δ_r and Δ_R

$$\begin{aligned} F =& (-k_\mathrm{r}e_\mathrm{r} - k_\mathrm{v}e_\mathrm{v} - mge_3 + m\ddot{r}_\mathrm{d} - \eta_\mathrm{r} + \mu_\mathrm{r})^\top Re_3 \\ \tau =& -k_\mathrm{R}e_\mathrm{R} - k_\Omega e_\Omega + \omega \times J\omega \\ & - J\big(\hat{\Omega}R^\top R_\mathrm{d}\Omega_\mathrm{d} - R^\top R_\mathrm{d}\dot{\Omega}_\mathrm{d}\big) - \eta_\mathrm{R} + \mu_\mathrm{R} \end{aligned} \tag{14a}$$

where $k_\mathrm{r}, k_\mathrm{v}, k_\mathrm{R}, k_\omega \in \mathbb{R}$ are the controller gains and

$$e_\mathrm{r} = r - r_\mathrm{d}, \qquad e_\mathrm{R} = \frac{1}{2}\big(R_\mathrm{d}^\top R - R^\top R_\mathrm{d}\big)^\vee \tag{15a}$$

$$e_\mathrm{v} = \dot{r} - \dot{r}_\mathrm{d}, \qquad e_\omega = \omega - R^\top R_\mathrm{d}\omega_\mathrm{d} \tag{15b}$$

are error terms with r_d, R_d, and ω_d corresponding to the position, orientation, and angular velocity references, $\mathrm{tr}(\cdot)$ is the trace operator, and the *vee operator* $(\cdot)^\vee : \mathrm{SO}(3) \to \mathbb{R}^3$ is the inverse of the hat operator $\hat{(\cdot)}$. The attitude tracking error e_R is interpreted as the gradient of the attitude error function characterized by [2]

$$\Psi(R, R_\mathrm{d}) = \frac{1}{2}\mathrm{tr}\big(I - R_\mathrm{d}^\top \mathrm{R}\big). \tag{16}$$

Furthermore, the angular velocity error term satisfies the equation $\dot{\Psi} = e_\mathrm{R}^\top e_\omega$.

We identify the external disturbances from noisy observations using GPs, more specifically in the form

$$\hat{\Delta}_\mathrm{r} = \mathcal{GP}_\mathrm{r}(\mathcal{X}) \sim \mathcal{N}(\eta_\mathrm{r}(\mathcal{X}), \Sigma_\mathrm{r}(\mathcal{X})) \tag{17a}$$

$$\hat{\Delta}_\mathrm{R} = \mathcal{GP}_\mathrm{R}(\mathcal{X}) \sim \mathcal{N}(\eta_\mathrm{R}(\mathcal{X}), \Sigma_\mathrm{R}(\mathcal{X})). \tag{17b}$$

The mean of GP is then directly used in (14) to compensate the effect of $\Delta_\mathrm{r}(\mathcal{X})$, $\Delta_\mathrm{R}(\mathcal{X})$. The uncertainty of the approximations, characterized by the covariances Σ_r and Σ_R, is handled by introducing the additional terms μ_r and μ_R that make the controller robust to this uncertainty. To define these terms, we assume that the GPs are trained until the true $\Delta_\mathrm{r}(\mathcal{X})$, $\Delta_\mathrm{R}(\mathcal{X})$ are inside the 95% confidence interval. We can now define μ_r

and μ_R similar to [9], as

$$\mu_r = -\frac{\delta_r^{\tau+2} e_B \|e_B\|^\tau}{\delta_r^{\tau+1}\|e_B\|^{\tau+1} + \epsilon_r^{\tau+1}}, \quad e_B = e_v + \frac{c_1}{m} e_r$$
$$\mu_R = -\frac{\delta_R^2 e_A}{\delta_R \|e_A\| + \epsilon_R}, \quad e_A = e_\Omega + c_2 J^{-1} e_R \tag{18}$$

where $c_1, c_2, \epsilon_r, \epsilon_R, \tau$ are positive constants, $\tau > 2$, δ_r, δ_R are the uncertainty bounds, and $\|\cdot\|$ is the Euclidean vector norm. However, instead of estimating δ_r, δ_R as in [9], we use the uncertainty of the corresponding trained GP. We calculate the standard deviation of a GP at an evaluation point as the norm of the square root of the covariance matrix (sphere bounding the ellipsoidal level set)

$$\sigma_r(\chi) = \|L_r(\chi)\|_2 = \sqrt{\|\Sigma_r(\chi)\|_2}, \quad L_r L_r^\top = \Sigma_r \tag{19a}$$
$$\sigma_R(\chi) = \|L_R(\chi)\|_2 = \sqrt{\|\Sigma_R(\chi)\|_2}, \quad L_R L_R^\top = \Sigma_R \tag{19b}$$

where $\|\cdot\|_2$ is the 2,2 induced norm (also called the spectral norm) of a matrix which is equal to its largest singular value. We define the uncertainty bounds using the 95% confidence interval of the normal distribution, namely,

$$\hat{\delta}_r(\chi) = 2\sigma_r(\chi), \quad \hat{\delta}_R(\chi) = 2\sigma_R(\chi). \tag{20}$$

From the confidence interval, we calculate an ultimate bound for the difference between the disturbances and the adaptive terms over the operating domain as follows:

$$\delta_r = \max_{\chi} \hat{\delta}_r(\chi), \quad \delta_R = \max_{\chi} \hat{\delta}_R(\chi). \tag{21}$$

Now we can show for the considered uncertainty bound (21) that the following stability guarantee holds true.

Theorem 1: Consider that the control force F and torque τ defined by (14) are applied on the uncertain system (13). Given any $\psi_{1,\max}, e_{r_{\max}} > 0$ and initial conditions that satisfy

$$\Psi(R(0), R_d(0)) < \psi_{1,\max} < 1 \tag{22a}$$
$$\|e_r(0)\| < e_{r_{\max}} \tag{22b}$$

then there exists a controller (in terms of the choice of parameters k_r, k_v, k_R, k_ω, c_1, c_2, ϵ_r, ϵ_R, τ) such that all the error terms in (15) are uniformly ultimately bounded.

Proof: Throughout the proof, we adapt the steps of Proposition 3 in [9] to the GP-based uncertainty bounds. First, we derive the dynamics of the rotational and translational tracking error and construct Lyapunov functions for them.

Based on (13), (14), and (15), the error dynamics can be expressed in the following form:

$$m\dot{e}_v = mge_3 - m\ddot{r}_d - \frac{F}{e_3^\top R_d^\top R e_3} R_d e_3 - X + \Delta_r$$
$$= -k_r e_r - k_v e_v - X + \Delta_r - \eta_r + \mu_r \tag{23a}$$
$$X = \frac{F}{e_3^\top R_d^\top R e_3}((e_3^\top R_d^\top R e_3) R e_3 - R_d e_3) \tag{23b}$$
$$J\dot{e}_\omega = \tau + \Delta_R - \omega \times J\omega + J\left(\hat{\omega} R^\top R_d \omega_d - R^\top R_d \dot{\omega}_d\right)$$
$$= -k_R e_R - k_\omega e_\omega + \Delta_R - \eta_R + \mu_R. \tag{23c}$$

Similar to [9], consider the following Lyapunov function candidates:

$$\mathcal{V}_1 = \frac{1}{2} k_r \|e_r\|^2 + \frac{1}{2} m \|e_v\|^2 + c_1 e_r^\top e_v \tag{24a}$$
$$\mathcal{V}_2 = \frac{1}{2} e_\omega^\top J e_\omega + k_R \Psi(R, R_d) + c_2 e_R^\top e_\omega. \tag{24b}$$

First, let us focus on $\mathcal{V}_2$. For the attitude error function Ψ, the lower and upper bounds can be given in terms of the attitude error e_R as follows:

$$\frac{1}{2}\|e_R\|^2 \le \Psi(R, R_d) \le \frac{1}{2 - \psi_{1,\max}} \|e_R\|^2. \tag{25}$$

The detailed derivation of these bounds can be found in [24]. Note that (25) implies that Ψ is positive definite and decrescent for all $t \in \mathbb{R}_+$. Using (25), the following upper bound can be derived for the time derivative of $\mathcal{V}_2$:

$$\dot{\mathcal{V}}_2 \le -z_2^\top W_2 z_2 + e_A^\top (\Delta_R - \eta_R + \mu_R)$$
$$z_2 = \begin{bmatrix} \|e_R\| \\ \|e_\omega\| \end{bmatrix} \in (\mathbb{R}^2)^{\mathbb{R}_+}, \quad W_2 = \begin{bmatrix} \frac{c_2 k_R}{\lambda_M} & -\frac{c_2 k_\omega}{2\lambda_m} \\ -\frac{c_2 k_\omega}{2\lambda_m} & k_\omega - c_2 \end{bmatrix} \tag{26}$$

where λ_m, λ_M denote the smallest and largest eigenvalues of the inertia matrix J, respectively. If the controller parameters are chosen such that W_2 is positive definite, then $z_2^\top W_2 z_2$ is a positive definite function. Now we examine the second term on the right-hand side of (26) and construct an upper bound for this term as well. First, note that $\|\Delta_R - \eta_R\| \le \delta_R$ holds under the assumption that Δ_r, Δ_R are inside the 95% confidence interval of the GP distribution. Using this inequality and the definition of the robust control law (18), the following upper bound can be obtained:

$$e_A^\top(\Delta_R - \eta_R + \mu_R) \le \delta_R \|e_A\| - \frac{\delta_R^2 \|e_A\|^2}{\delta_R \|e_A\| + \epsilon_R}$$
$$= \frac{\delta_R \|e_A\|}{\delta_R \|e_A\| + \epsilon_R} \epsilon_R \le \epsilon_R \tag{27}$$

where ϵ_R is chosen to be a sufficiently small positive constant, and therefore,

$$\dot{\mathcal{V}}_2 \le -z_2^\top W_2 z_2 + \epsilon_R. \tag{28}$$

The right-hand side of (28) is a shifted negative definite function, and hence, the tracking errors e_R and e_ω are uniformly ultimately bounded.

Consider now Lyapunov function candidate $\mathcal{V}_1$. Its time derivative can be given as follows:

$$\dot{\mathcal{V}}_1 = -(k_v - c_1)\|e_v\|^2 - \frac{c_1 k_r}{m}\|e_r\|^2 - \frac{c_1 k_v}{m} e_r^\top e_v$$
$$+ (X + \Delta_r - \eta_r + \mu_r)^\top \left(\frac{c_1}{m} e_r + e_v\right). \tag{29}$$

Similar to the case of $\mathcal{V}_2$, the first three terms can be made negative definite by suitably choosing the parameters of the controller. The last term can be divided into two parts: the first one is $e_B^\top(\Delta_r - \eta_r + \mu_r)$, and $e_B^\top X$ is the second. Using (21) and (18), an upper bound can be derived for the first term

$$e_B^\top(\Delta_r - \eta_r + \mu_r) \le \delta_r \|e_B\| - \frac{\delta_r^{\tau+2}\|e_B\|^{\tau+2}}{\delta_r^{\tau+1}\|e_B\|^{\tau+1} + \epsilon_r^{\tau+1}}$$
$$= \frac{\delta_r \|e_B\|}{\delta_r^{\tau+1}\|e_B\|^{\tau+1} + \epsilon_r^{\tau+1}} \epsilon_r^{\tau+1} \le \epsilon_r \tag{30}$$

where similar to ϵ_R, ϵ_r is also a sufficiently small positive constant, both typically in the order of magnitude of 10^{-4}–10^{-2}. To obtain an upper bound for $e_B^\top X$, first we construct an upper bound for X using (23b)

$$\|X\| \le \|A\| \|(e_3^\top R_d^\top R e_3) R e_3 - R_d e_3\|$$
$$\le (k_r\|e_r\| + k_v\|e_v\| + B + \delta_r)\|(e_3^\top R_d^\top R e_3) R e_3 - R_d e_3\| \tag{31}$$

where $A = -k_r e_r - k_v e_v - mge_3 + m\ddot{r}_d - \eta_r + \mu_r$, and we assume that the reference trajectory r_d has been designed such that condition $\| - mge_3 + m\ddot{r}_d - \eta_r\| < B$ holds for some $B > 0$.

Using the following relation (see [9] for details)

$$\left\|\left(e_3^\top R_\mathrm{d}^\top R e_3\right) R e_3 - R_\mathrm{d} e_3\right\| \leq \|e_\mathrm{R}\| = \sqrt{\Psi(2-\Psi)} \leq \left\{\sqrt{\psi_{1,\max}\left(2-\psi_{1,\max}\right)} \triangleq \alpha\right\} < 1 \tag{32}$$

the upper bound for X can be expressed as follows:

$$\|X\| \leq (k_\mathrm{r}\|e_\mathrm{r}\| + k_\mathrm{v}\|e_\mathrm{v}\| + B + \delta_\mathrm{r})\alpha. \tag{33}$$

By substituting (30) and (33) into (29), we obtain

$$\begin{aligned}\dot{\mathcal{V}}_1 \leq & -(k_\mathrm{v}(1-\alpha) - c_1)\|e_\mathrm{v}\|^2 - \frac{c_1 k_\mathrm{r}}{m}(1-\alpha)\|e_\mathrm{r}\|^2 \\ & + \|e_\mathrm{R}\|\left\{(B+\delta_\mathrm{r})\left(\frac{c_1}{m}\|e_\mathrm{r}\| + \|e_\mathrm{v}\|\right) + k_\mathrm{r}\|e_\mathrm{r}\|\|e_\mathrm{v}\|\right\} \\ & + \frac{c_1 k_\mathrm{v}}{m}(1+\alpha)\|e_\mathrm{r}\|\|e_\mathrm{v}\| + \epsilon_\mathrm{r}.\end{aligned} \tag{34}$$

According to the proof of Proposition 3 in [9], inequality (34) implies that the tracking errors e_r and e_v are uniformly ultimately bounded as well. This completes the proof. □

Remark: The user-specified tolerable error bounds $\psi_{1,\max}$, $e_{\mathrm{r}_\mathrm{max}}$ in (22) are required to be realistic to achieve high control performance in the allowed input range.

B. Trajectory Planning for the Flip Maneuver

To use the geometric tracking controller to perform the flip maneuver, a suitable reference trajectory is needed. For this purpose, we introduce an optimization-based trajectory design method which is an additional contribution of this article.

Based on the controller structure (14), first an attitude reference trajectory R_d is constructed and then it is completed with a position reference r_d. Similar to the feedforward approach, the objective for trajectory planning is that the quadcopter should arrive as close to the starting point as possible, while keeping the control inputs within the allowed range during the maneuver.

The attitude reference is specified in unit quaternions: $q_\mathrm{d} = [\ q_{\mathrm{d},0}\ q_{\mathrm{d},1}\ q_{\mathrm{d},2}\ q_{\mathrm{d},3}\ ]^\top$, where $q_{\mathrm{d},0}$ is the scalar part of the quaternion, and $q_{\mathrm{d},2}$ corresponds to the pitch angle, as $q_{\mathrm{d},1} = q_{\mathrm{d},3} = 0$, because both the roll and yaw angles are zero during the flip. Using that q_d is a unit quaternion, we can express the third element of it as $q_{\mathrm{d},2} = (1 - q_{\mathrm{d},0}^2)^{1/2}$, and hence it is sufficient to design a trajectory only for $q_{\mathrm{d},0}$. A 360° rotation around the y-axis means that the scalar part of the attitude quaternion goes from 1 to −1. In the trajectory design, it is important to stay within the $q_{\mathrm{d},0}(t) \in [-1, 1]$ range, because only unit quaternions describe rotation. For this purpose, the smooth sigmoid function

$$q_{\mathrm{d},0}(t) = \frac{2}{1 + e^{-\nu_\mathrm{m}\left(t - \frac{t_\mathrm{m}}{2}\right)}} - 1 \tag{35}$$

is chosen to describe the scalar part of the reference attitude, where the parameters are the horizontal scaling of the sigmoid curve ν_m and the execution time t_m. The attitude quaternion reference trajectory is displayed in Fig. 4. Assuming that $\phi \equiv \psi \equiv 0$ during the flip, the conversion to Euler angles yields $\theta = 2\arccos(q_{0,\mathrm{d}})$, where $\theta(t) \in [-\pi, \pi]$. Hence, the pitch angle goes smoothly from zero to π, jumps to $-\pi$, and goes smoothly to zero. Besides rotation, the maneuver also requires translational motion, because without proper lifting at the beginning, the quadcopter would fall to the ground due to gravity. The position reference is designed considering that the rotational and translational equations of the dynamical model are coupled. The translational motion of the flip maneuver is within the xz plane, and therefore, $y_\mathrm{d}(t) = 0$. The other two equations of the translational dynamics in (7a) are

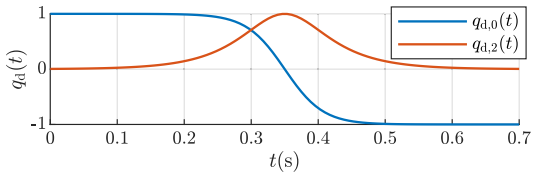


Fig. 4. Attitude quaternion reference trajectory for the backflip maneuver with $\nu_\mathrm{m} = 35$ 1/s and $t_\mathrm{m} = 0.7$ s.

$$m\ddot{x} = -F R_{1,3} \tag{36a}$$

$$m\ddot{z} = -F R_{3,3} + mg \tag{36b}$$

where $R_{i,j}$ denotes the (i, j)th entry of the rotation matrix R. However, assuming that the attitude rapidly converges to the reference, we can substitute the reference rotation matrix in (36), resulting in the translational state-space representation

$$\dot{\zeta} = A\zeta + Bu \tag{37}$$

$$\zeta = \begin{bmatrix} x \\ \dot{x} \\ \tilde{z} \\ \dot{\tilde{z}} \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \frac{1}{m}\begin{bmatrix} 0 \\ R_{\mathrm{d},1,3} \\ 0 \\ R_{\mathrm{d},3,3} \end{bmatrix}$$

where ζ is the reduced state vector capturing the translational motion, and $R_{\mathrm{d},i,j}$ are the corresponding elements of the reference rotation matrix R_d (converted from the reference quaternion q_d). As the motion equations are decoupled, the effect of gravity can be added to z which is denoted by $\tilde{z}$ (modified state) in the equation. Note that (37) is a *linear time-varying* (LTV) state-space representation with the thrust force $F = u$ as the only control input.

By discretizing (37) using complete, zero-order hold discretization with discretization step size $T_\mathrm{s} > 0$

$$\zeta_{k+1} = A_\mathrm{d}\zeta_k + B_{\mathrm{d},k}u_k$$

$$A_\mathrm{d} = \begin{bmatrix} 1 & T_\mathrm{s} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_\mathrm{s} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B_{\mathrm{d},k} = \frac{T_\mathrm{s}}{m}\begin{bmatrix} \frac{T_\mathrm{s}}{2}R_{\mathrm{d},1,3}(k) \\ R_{\mathrm{d},1,3}(k) \\ \frac{T_\mathrm{s}}{2}R_{\mathrm{d},3,3}(k) \\ R_{\mathrm{d},3,3}(k) \end{bmatrix} \tag{38}$$

where $k \in \mathbb{Z}$ denotes the discrete time, i.e., ζ_k expresses $\zeta(kT_\mathrm{s})$, a quadratic programming problem can be formulated over a finite horizon, similar to model predictive control, to find a motion trajectory for executing the flip. The input of the model is the collective thrust of the propellers, $u_k = F_k$. For a fixed duration of the maneuver with N discrete time steps, the following quadratic optimization problem is formulated:

$$\begin{aligned} \min_{\{u_k\}_{k=1}^N} \ & \sum_{k=1}^{N}\left[\left(\zeta_k - \zeta_{\mathrm{d},k}\right)^\top Q_k\left(\zeta_k - \zeta_{\mathrm{d},k}\right) + u_k^\top W_k u_k\right] \\ \text{s.t.} \ & \zeta_{k+1} = A_\mathrm{d}\zeta_k + B_{\mathrm{d},k}u_k \\ & \{\zeta_k\}_{k=1}^N \in \mathcal{X}, \quad \{u_k\}_{k=0}^N \in \mathcal{U} \end{aligned} \tag{39}$$

TABLE I
CONSIDERED PHYSICAL PARAMETERS OF THE CRAZYFLIE 2.1

Mass	m	28 g
Prop-to-prop length	l	92 mm
Diagonal inertia	J_{xx}	$1.4 \cdot 10^{-5}$ kgm^2
	J_{yy}	$1.4 \cdot 10^{-5}$ kgm^2
	J_{zz}	$2.17 \cdot 10^{-5}$ kgm^2
Thrust coefficient	c	$2.88 \cdot 10^{-8}$ Ns^2
Drag coefficient	b	$7.24 \cdot 10^{-10}$ Nms^2

where $Q_k \in \mathbb{R}^{4\times4}$ and $W_k \in \mathbb{R}$ are the weight matrices, ζ_0 is the initial state, and $\mathcal{X}, \mathcal{U}$ are constraint sets for the states and the control input, respectively. The only objective of the trajectory design is to minimize the final position error of the quadcopter and keep the position within a specified range, and therefore, the weight matrices are $(W_k, Q_k) = (0, 0)$ for $k = 1, \ldots, N-1$, except for the weight of the final state that is $Q_N = \text{diag}(1, 0, 1, 0)$ while $W_N = 0$. As all the other weights are zero, it is only required to define a final state position reference $\zeta_{\text{d},N}$, the components of which are zero except for the effect of the gravity in $\tilde{z}_{\text{d},N} = 0.5 T_\text{s}^2 N^2 g$.

We specify linear constraints for the states: $x \in [x_-, x_+]$, $z \in [z_-, z_+]$ to model the available space for the maneuver, preventing collisions with other objects or walls. We also define linear constraints for the control input, namely,

$$\frac{\|\tau_k\|}{l} \leq u_k = F_k \leq F_\text{max} - \frac{\|\tau_k\|}{l} \tag{40}$$

where τ_k is the vector of the three torques around the three body axes, out of which $\tau_{x,k} = \tau_{z,k} = 0$ normally during the flip, l is the distance of the quadcopter center of mass and the propellers projected to the xz plane, and F_max is the maximal collective thrust of the rotors. The torque control input τ_k is calculated from the reference attitude R_d based on (14a) assuming that the rotation errors e_R, e_ω are zero.

The optimization problem in (39) can be solved easily using an off-the-shelf QP solver, e.g., by `quadprog` in MATLAB. Finally, to get a smooth trajectory, we fit cubic splines on the discrete reference points $\{\zeta_k\}_{k=0}^N$ obtained in (39).

VI. SIMULATION STUDY

A. Environment

To analyze the properties and the performance of the introduced methods, we tested them in a simulation environment based on the dynamic model of a Bitcraze Crazyflie 2.1 miniature quadcopter. The same drone is used in real experiments, presented in Section VII. For both simulation and control design, the considered physical parameters of the quadcopter are given in Table I, which are based on [25]. The simulations have been executed using MuJoCo, a high-fidelity physics engine.[2] The implementation of GP regression is based on GPyTorch [26], while the Bayesian Optimization is based on [27]. All the simulation code used in this work is available at our GitHub,[3] and a video illustrating the simulation results can be found at https://youtu.be/Ed9jYlZr95c. In this section

[2]https://mujoco.org/
[3]https://github.com/AIMotionLab-SZTAKI/crazyflie_backflipping

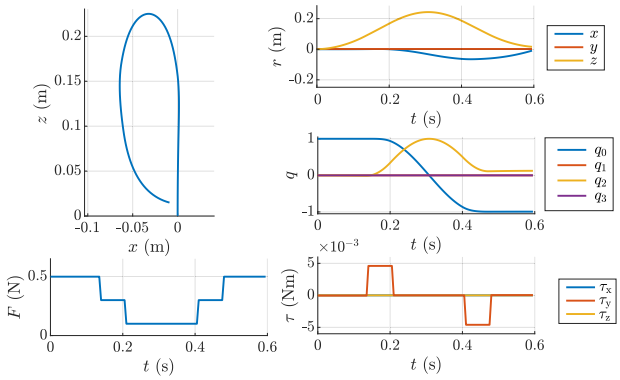


Fig. 5. Backflipping in simulation by feedforward control. The position (r), orientation (q), collective thrust (F), and input torque (τ) are displayed.

and in Section VII, we display obtained results with the z-axis pointing upward (in contrast to the NED convention discussed in Section III), because the backflip maneuver is more illustrative this way.

B. Bayesian Optimized Feedforward Control

The motion primitive parameters of the backflip maneuver have been calculated as the solution of optimization (10), using 250 random initial function evaluations and 1000 iterations. The result of the numerical optimization is

$$\begin{aligned} p^* &= \left[U_1^* \; t_1^* \; t_3^* \; U_5^* \; t_5^*\right]^\top \\ &= \left[17.8 \; 0.14 \; 0.2 \; 17.8 \; 0.12\right]^\top \end{aligned} \tag{41}$$

where the unit of the collective accelerations U_i^* is m/s^2, and the time is in seconds. The simulation results are displayed in Fig. 5, using the optimized parameter vector given by (41). On the left plot, the position of the quadcopter during the flip is shown. At the end of the optimal maneuver, the position is $r = [-0.009, 0, 0.015]$ m with $[\phi, \theta, \psi] = [0, 0.24, 0]$ rad orientation, and thus the elements of the final state error e are uniformly small. On the right, the trajectory of the position vector, the orientation in quaternion representation, the collective thrust, and the input torques are shown, where the five phases of the maneuver defined in Section IV can be clearly identified. The figure shows that almost near-maximal and near-minimal collective thrust and torque commands are required to perform the maneuver successfully. Switching between these extreme values introduces discontinuities of the control input where the unmodeled transient behavior of the actuator dynamics can be significant. The latter can influence the performance of the control strategy. A possible solution would be to use a different parametrization of the control inputs instead of bang-bang control (e.g., spline parameters); however, such changes would make the feedforward design much more complex.

C. Geometric Control and Trajectory Planning

The second approach to perform a flip maneuver is trajectory planning and reference tracking with geometric control. Based on the results of the flip with feedforward control, the parameters of the reference pitch trajectory are chosen to be

$v_{\mathrm{m}} = 35$ 1/s and $t_{\mathrm{m}} = 0.7$ s as illustrated in Fig. 4. The quadratic optimization in (39) is solved under the following constraints:

$$\mathcal{X}: \ \{x_-, x_+, z_-, z_+\} = \{-0.15, 0, 0, 0.3\} \text{ m}$$
$$\mathcal{U}: \ F \in [0, 0.64] \text{ N}$$

with sampling time $T_{\mathrm{s}} = 2$ ms. Based on [25], the maximal collective thrust limit is chosen to be $F_{\max} = 0.64$ N together with position bounds such that the trajectory is feasible and the quadcopter exploits the available flying space while avoiding collision with walls and obstacles. The quadratic optimization in (39) is solved off-line before starting the maneuver, on a desktop PC with Intel Core i9 processor and 16 GB of RAM. The computation time of the trajectory is 0.11 s in average, using MATLAB with Mosek.[4]

The gains of the geometric controller have been determined based on the stability conditions detailed in [2], resulting in

$$k_{\mathrm{r}} = 4.5, \quad k_{\mathrm{v}} = 0.3, \quad k_{\mathrm{R}} = 0.2, \quad k_{\omega} = 0.002. \tag{42}$$

The simulation results of the trajectory planning and reference tracking with nominal geometric control are displayed in Figs. 6 and 7. The trajectory of the control torque is smooth compared with the commands given by the feedforward controller, resulting in less possible sensitivity with respect to unmodeled actuator dynamics. Moreover, Fig. 7 shows that highly accurate tracking of the attitude reference can be achieved (Ψ has the order of magnitude of 10^{-5}), which also leads to small position tracking errors.

Next, we evaluate the performance of the proposed robust adaptive geometric controller for backflipping. We assume that the most significant modeling errors compared with the real drone arise in the rotational dynamics, due to the inaccuracy of the inertia matrix and center of gravity, and other aerodynamic effects. In simulation, we apply an external disturbance characterized by

$$\Delta_{\mathrm{R}} = \begin{bmatrix} -0.007 & -0.007 & 0 \end{bmatrix}^{\top} \cdot \sin\left(\frac{\phi}{2} + \frac{\theta}{2}\right) \text{ Nm}. \tag{43}$$

We use the controller gains given by (42) and choose the following constant values based on the stability conditions detailed in [9]: $\tau = 3, c_1 = 1, c_2 = 0.1, \epsilon_r = \epsilon_R = 0.0004$.

In the example of the backflip maneuver, we use only two scalar adaptive terms: the roll and pitch terms of η_{R}, namely, $\eta_{\mathrm{R},1}$ and $\eta_{\mathrm{R},2}$, because most of the uncertainties arise in the roll and pitch motions. The adaptive terms are represented by two independent GPs with the following 4-D input: the x and y elements of the attitude quaternions (q_1, q_2) and the angular velocity elements ω_x and ω_y. The adaptive GP terms introduced in Section V depend on the full state vector; however, in case of the backflip scenario, only these four inputs are relevant, and by reducing the input dimension, the model complexity is decreased radically without losing its expressiveness. For the GPs, we use zero mean and SE covariance, given by (2). The signal variance and lengthscale hyperparameters are trained using maximum likelihood estimation on 125 training points. Due to the relatively small input dimension and number of training points, the training and evaluation of the GPs are fast and efficient.

[4]https://www.mosek.com/

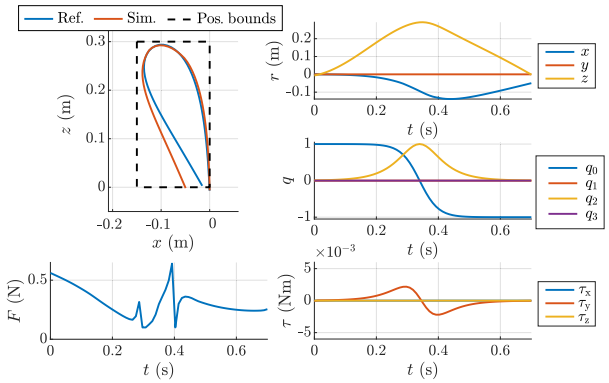


Fig. 6. Backflipping in simulation by geometric control. The position, orientation, and control inputs are displayed.

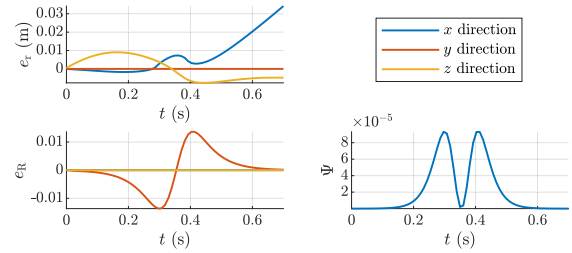


Fig. 7. Trajectory of the error terms e_{r}, e_{R} and the attitude error function Ψ in simulation by geometric control.

The simulations by robust adaptive geometric control are shown in Figs. 8 and 9. We compare the results using nominal geometric control (without adaptive and robust terms), geometric control with GP mean (without robust terms), and robust adaptive geometric control given by (14). Our results show that the attitude error of the nominal controller during backflipping is reduced significantly using the proposed solutions, especially in terms of e_{R_1}, e_{R_3}. The position error is most significant in the x-direction (e_{r_1} is an order of magnitude larger than $e_{\mathrm{r}_2}, e_{\mathrm{r}_3}$), where the adaptive and robust controllers are capable to radically improve the tracking performance.

VII. Experimental Study

A. Experimental Setup

The real experiments are performed with a Bitcraze Crazyflie 2.1 drone. Optitrack motion capture system[5] is used to provide high-precision position and orientation information. The drone and the positioning system are interconnected via a ground control PC, which runs the high-level experiment management and executes data-logging as well. The block diagram presenting the interconnection of the components is shown in Fig. 10. The quadrotor is equipped with an IMU containing a 3-D accelerometer, gyroscope, magnetometer, and barometer, and it has two microcontrollers: an STM32F405 for running the flight controller and an nRF51822 for radio communication and power management. In addition, we use an expansion deck for high-speed logging of measurement data to a micro SD-card. The quadcopter runs the original Bitcraze firmware augmented with our proposed control algorithms, while on the server, the Crazyswarm software platform is used

[5]https://optitrack.com

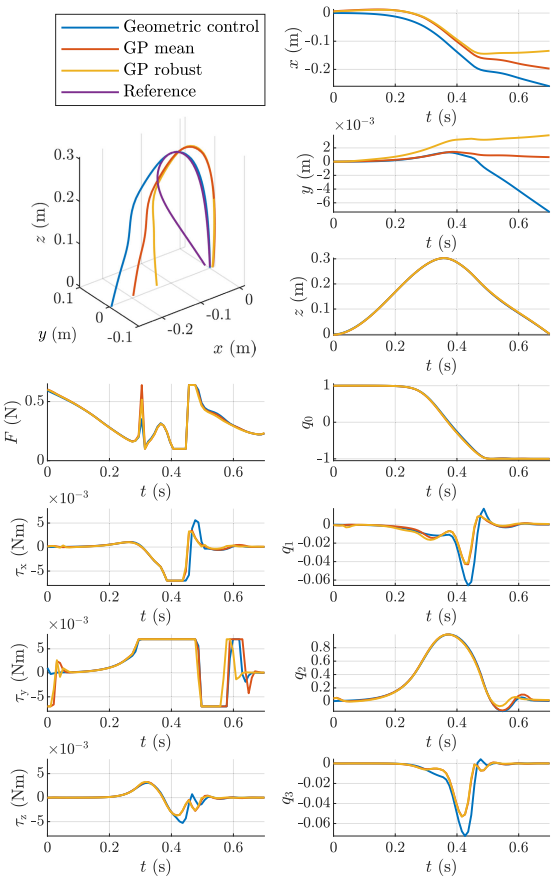


Fig. 8. Backflipping in simulation using nominal, adaptive, and robust geometric control, with additional uncertainty. The position, orientation, and control inputs are displayed.

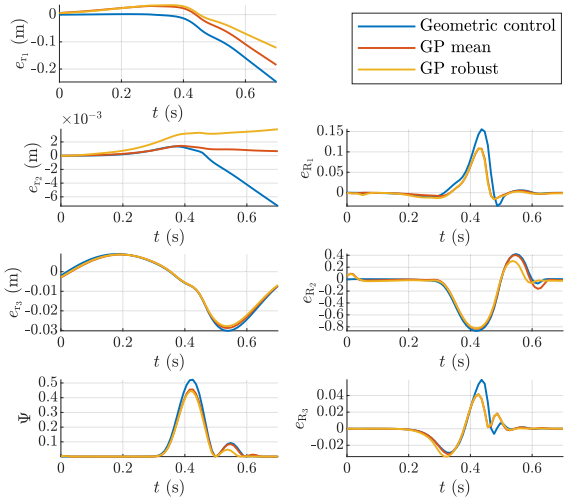


Fig. 9. Comparison of the error terms $e_{\mathrm{r}}, e_{\mathrm{R}}$ and the attitude error function Ψ in simulation with additional uncertainty.

to ease the implementation and configuration of high-level control components [28].

B. Optimized Feedforward Control

First, we evaluate the results of performing the backflip with optimization-based feedforward control. For this experiment, the optimized parameter set given by (41) is used.

The measurement results are displayed in Fig. 11, showing that the flip is executed with $[x\ y\ z] = [-0.22\ 0.008\ 0.01]$ m final position error and $[\phi\ \theta\ \psi] = [0.032\ 0.41\ 0.021]$ rad final error in Euler angles. Compared with the simulation displayed in Fig. 5, the maximal displacement from the origin in direction x is around four times larger, while in direction z is around 23% larger. The difference is due to the uncertainties of the simulation model; however, the backflip is still performed successfully and the quadcopter is near the initial

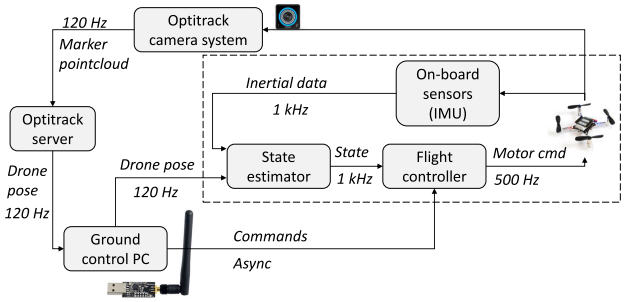


Fig. 10. Block diagram of the experimental setup: indoor quadcopter navigation with internal and external localization.

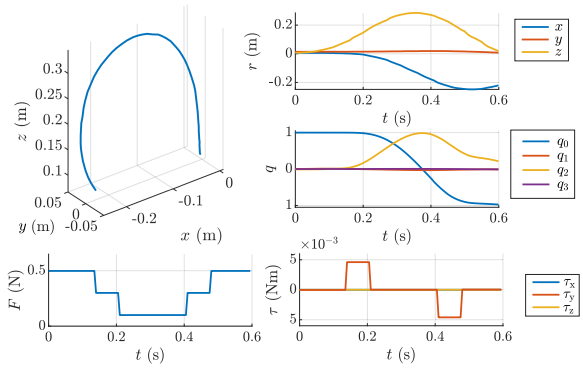


Fig. 11. Backflipping measurement results with feedforward control using the optimized parameter vector given by (41). The position, orientation, and control inputs are displayed.

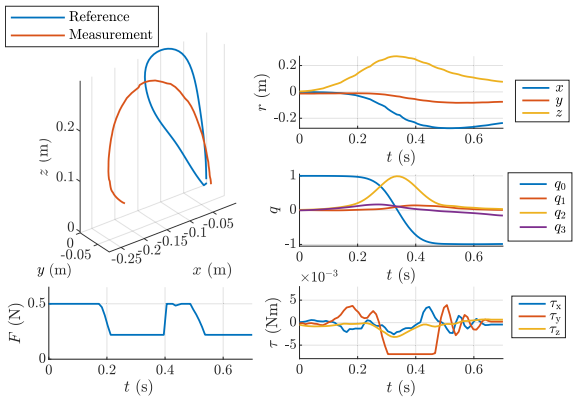


Fig. 12. Backflipping measurement results with nominal geometric control. The position, orientation, and control inputs are displayed.

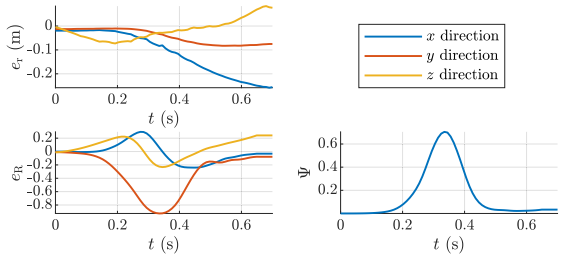


Fig. 13. Error terms $e_{\mathrm{r}}, e_{\mathrm{R}}$ and the attitude error function Ψ in real experiments by geometric control.

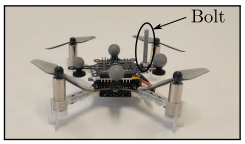


Fig. 14. Crazyflie 2.1 quadcopter with reflective markers and a steel bolt used to introduce significant additional dynamics.

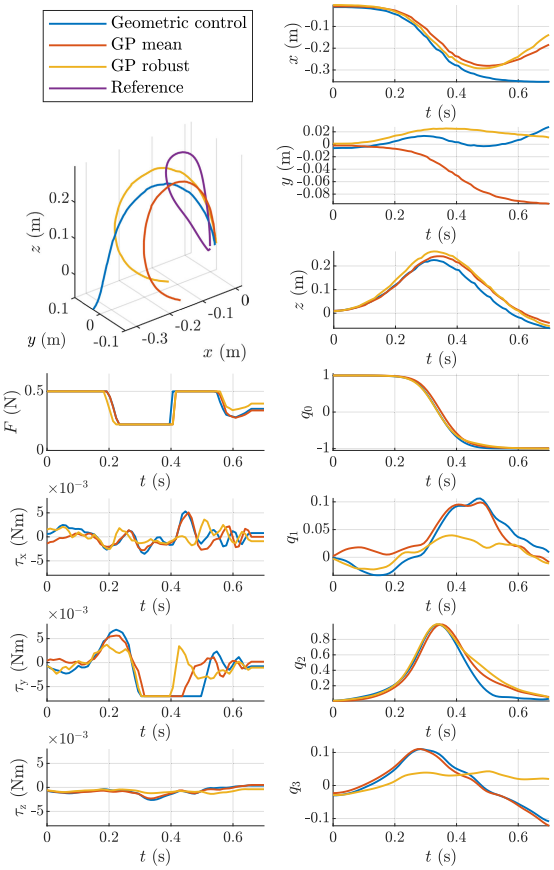


Fig. 15. Measurement results with a bolt attached to the quadcopter, using nominal, adaptive, and robust geometric control. The position, orientation, and control inputs are displayed.

configuration at the end of the maneuver. It is important to note that the feedforward approach is sensitive to uncertainties in the dynamics and initial conditions. For example, if the flip maneuver begins when the orientation of the quadcopter is not horizontal, the stability can be lost at the recovery phase.

C. Trajectory Planning and Geometric Control

The experimental results of backflipping with the nominal geometric control are displayed in Figs. 12 and 13. The most important part of reference tracking is the attitude error function Ψ and error vector e_R, because a fast, stable, and accurate attitude tracking is required to perform the flip maneuver and recover successfully. As shown in the left plot of the measurement results, the attitude error e_R is small in all the directions and the controller remains stable. Although the position error e_r is larger (especially in the x-direction), the stability of the controller guarantees that the quadcopter gets back to the initial position after the backflip maneuver. In spite of the imperfect position tracking, the geometric controller is able to perform the backflip maneuver exactly the same way ten out of ten times, which indicates that even the nominal

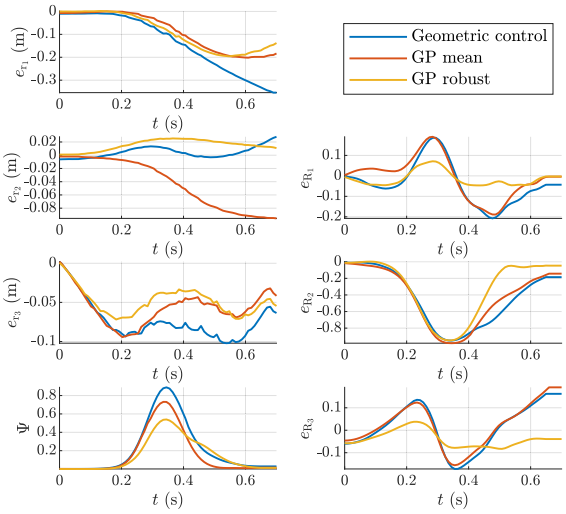
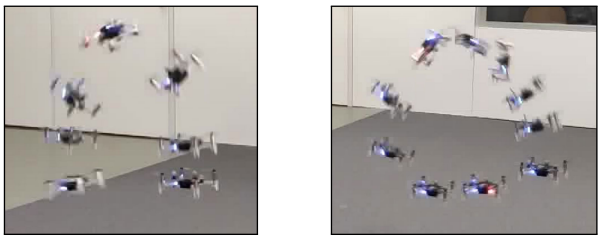


Fig. 16. Trajectory of the error terms e_r, e_R and the attitude error function Ψ in real experiments with a bolt attached to the quadcopter.



(a) (b)

Fig. 17. Composite images of the experiments using a Bitcraze Crazyflie 2.1 quadcopter. (a) Optimization-based feedforward control. (b) Robust geometric feedback control.

geometric control algorithm is significantly more robust than the feedforward method.

Next, we evaluate the performance of the proposed robust adaptive geometric controller with model uncertainty added to the quadcopter dynamics.[6] We have found that a steel bolt attached to the drone (as illustrated in Fig. 14) has significant influence on the attitude dynamics; however, the vehicle is still able to perform the backflip maneuver. Using the modified configuration, we collect data points by performing agile maneuvers with the nominal geometric controller and fit a GP on the measurement data. During the flights, the GP has to be evaluated at 500 Hz, which is not possible due to the limited computational capacity of the on-board microcontroller unit. Therefore, we generate a lookup table by evaluating the trained GP on a grid of the input variables with 2025 grid points off-line and upload it to the on-board microcontroller of the quadcopter. Another viable alternative would be the use of sparse GP methods, e.g., [12], [19], which are possible to evaluate real-time, especially on larger quadcopter platforms with more powerful computational unit (e.g., NVIDIA Jetson [29]).

The measurement results are displayed in Figs. 15 and 16. Our results show that using the proposed adaptive and robust controllers, the attitude tracking error is even more reduced in real flights than it is in simulation. Moreover, the position tracking performance is also enhanced by the robust approach,

[6]The proposed controllers have shown similar performance and robustness for various trajectories with longer time duration; however, here only the results for the backflip maneuver are detailed.

especially in the x- and z-directions. The flight experiments are illustrated in Fig. 17, which shows composite images of the backflip maneuver.

VIII. Conclusion

In this article, a Bayesian-optimization-based feedforward control and a robust geometric reference tracking control approach with optimization-based trajectory planning have been proposed for quadcopters to reliably perform the backflip maneuver in the presence of modeling uncertainties. In the former, relatively simple approach, Bayesian optimization can be used to fine-tune the feedforward sequence, leading to efficient implementation both in simulation and real flights. The second method uses GP-based augmented motion models that are able to precisely approximate model uncertainties. Combined with geometric control, the resulting architecture provides robust stability and high control performance, outperforming the feedforward method both in terms of reliability and optimal tracking of the motion profile.

In our future research, we intend to use learning methods to perform complex maneuvers with less expert knowledge and extend the capabilities of the miniature drones even more.

References

[1] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Sci. Robot.*, vol. 6, no. 59, Oct. 2021.

[2] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. 49th IEEE Conf. Decis. Control (CDC)*, Dec. 2010, pp. 5420–5425.

[3] A. A. El-Badawy and M. A. Bakr, "Quadcopter aggressive maneuvers along singular configurations: An energy-quaternion based approach," *J. Control Sci. Eng.*, vol. 2016, pp. 1–10, 2016.

[4] Y. Chen and N. O. Pérez-Arancibia, "Lyapunov-based controller synthesis and stability analysis for the execution of high-speed multi-flip quadrotor maneuvers," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 3599–3606.

[5] P. Abbeel, A. Coates, and A. Y. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *Int. J. Robot. Res.*, vol. 29, no. 13, pp. 1608–1639, Nov. 2010.

[6] E. Kaufmann, A. Loquercio, R. Ranftl, M. Mueller, V. Koltun, and D. Scaramuzza, "Deep drone acrobatics," in *Proc. Robot., Sci. Syst.*, 2020. [Online]. Available: https://portal.issn.org/resource/ISSN/2330-765X and https://www.roboticsproceedings.org/

[7] S. Lupashin, A. Schoöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 1642–1648.

[8] P. Antal, T. Péni, and R. Tóth, "Nonlinear control method for backflipping with miniature quadcopters," *IFAC-PapersOnLine*, vol. 55, no. 14, pp. 133–138, 2022.

[9] T. Lee, M. Leok, and N. H. McClamroch, "Nonlinear robust tracking control of a quadrotor UAV on SE(3)," *Asian J. Control*, vol. 15, no. 2, pp. 391–408, Mar. 2013.

[10] F. A. Goodarzi, D. Lee, and T. Lee, "Geometric adaptive tracking control of a quadrotor unmanned aerial vehicle on SE(3) for agile maneuvers," *J. Dyn. Syst., Meas., Control*, vol. 137, no. 9, pp. 1–13, Sep. 2015.

[11] M. Bisheban and T. Lee, "Geometric adaptive control with neural networks for a quadrotor in wind fields," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 4, pp. 1533–1548, Jul. 2021.

[12] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[13] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 6, pp. 2736–2743, Nov. 2020.

[14] M. Liu, G. Chowdhary, B. Castra da Silva, S.-Y. Liu, and J. P. How, "Gaussian processes for learning and control: A tutorial with examples," *IEEE Control Syst. Mag.*, vol. 38, no. 5, pp. 53–86, Oct. 2018.

[15] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 2, pp. 408–423, Feb. 2015.

[16] G. Torrente, E. Kaufmann, P. Fohn, and D. Scaramuzza, "Data-driven MPC for quadrotors," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3769–3776, Apr. 2021.

[17] L. Bauersfeld, E. Kaufmann, P. Foehn, S. Sun, and D. Scaramuzza, "Neurobem: Hybrid aerodynamic quadrotor model," 2021, *arXiv:2106.08015*.

[18] M. Abdar et al., "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Inf. Fusion*, vol. 76, pp. 243–297, Dec. 2021.

[19] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1935–1959, Dec. 2005.

[20] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 20–32, Sep. 2012.

[21] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton, NJ, USA: Princeton Univ. Press, 2012.

[22] P. Frazier, "A tutorial on Bayesian optimization," 2018, *arXiv:1807.02811*.

[23] E. Brochu, V. Cora, and N. Freitas, "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," 2010, *arXiv:1012.2599*.

[24] T. Fernando, J. Chandiramani, T. Lee, and H. Gutierrez, "Robust adaptive geometric tracking controls on SO(3) with an application to the attitude dynamics of a quadrotor UAV," in *Proc. 50th IEEE Conf. Decision Control Eur. Control Conf.*, 2011, pp. 7380–7385.

[25] J. Förster, "System identification of the crazyflie 2.0 nano quadrocopter," Bachelor Thesis, Dept. Mech. Process Eng., ETH Zürich, Zürich, Switzerland, 2015.

[26] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1–11.

[27] F. Nogueira. (2014). *Bayesian Optimization: Open Source Constrained Global Optimization Tool for Python*. [Online]. Available: https://github.com/fmfn/BayesianOptimization

[28] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *Proc. Int. Conf. Robot. Autom.*, 2017, pp. 3299–3304.

[29] P. Foehn et al., "Alphapilot: Autonomous drone racing," *Auto. Robots*, vol. 46, pp. 307–320, Oct. 2022.