AL-TUNE: A Family of Methods to Effectively Tune UAV Controllers in In-flight Conditions

Dariusz Horla¹ · Wojciech Giernacki¹ · Tomas Báča² · Vojtech Spurny² · Martin Saska²

Received: 24 June 2020 / Accepted: 2 June 2021 / Published online: 3 August 2021
© The Author(s) 2021

Abstract

In the paper, a family of novel real-time tuning methods for an unmanned aerial vehicle (UAV) altitude controller in in-flight conditions. The methods allow the controller's gains to be adapted only on the basis of measurements from a basic sensory equipment and by constructing the optimization cost function in an on-line fashion with virtually no impeding computational complexity; in the case of the altitude controller as in this paper for a hexacopter, altitude measurements were used only. The methods are not dependent on the measurement level, and present the approach in a generally applicable form to tuning arbitrary controllers with low number of parameters. Real-world experimental flights, preceded by simulation tests, have shown which method should behave best in a noisy environment when e.g. wind disturbances act on a UAV while it is in autonomous flight. As the methods can potentially be extended to other control loops or controller types, making this a versatile, rapid-tuning tool. It has been shown that a well-tuned controller using the proposed AL-TUNE scheme outperforms controllers that are tuned just to stabilize the system. AL-TUNE provides a new way of using UAVs in terms of adaptivity to changing their dynamic properties and can be deployed rapidly. This enables new applications and extends the usability of fully autonomous UAVs, unlike other tuning methods, which basically require the availability of a UAV model. The core difference with respect to other research from the field is that other authors either use a model of a UAV to optimize the gains analytically or use machine learning techniques, what increases time consumption, whereas the presented methods offer a rapid way to tune controllers, in a reliable way, with deterministic time requirements.

Keywords Auto-tuning · UAV · Optimal controller gains · Altitude controller · In-flight tuning

✉ Dariusz Horla
dariusz.horla@put.poznan.pl

Wojciech Giernacki
wojciech.giernacki@put.poznan.pl

Tomas Báča
tomas.baca@fel.cvut.cz

Vojtech Spurny
vojtech.spurny@fel.cvut.cz

Martin Saska
martin.saska@fel.cvut.cz

¹ Faculty of Automatic Control, Robotics and Electrical Engineering, Institute of Robotics and Machine Intelligence, Poznan University of Technology, ul. Piotrowo 3a, 60-965 Poznan, Poland

² Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Karlovo namesti 13, 121 35 Prague 2, Czech Republic

1 Introduction

1.1 Motivation

Precise tuning of a controller determines the on-flight properties of any UAV architecture. High-quality tracking usually requires the use of nonlinear controllers that need information about the dynamics of the UAV. A model of UAV is therefore needed. This usually involves the use of high computational powers on-board, unlike the approach proposed here. As a solution, fixed-parameter PD-type or PID-type controllers can be used (see [26, 33]). However, these require proper tuning.

Well-tuned fixed-parameter controller gains result in high performance in UAV applications, where there is no need to perform any real-time adaptation. However, in aerial manipulation applications [6, 41], such as gripping elements and lifting them from the ground (see Fig. 1), the dynamics of the UAV change, and the controller parameters need to be altered. In many UAV missions, precise altitude

Fig. 1 Examples of the deployment of our system of autonomous UAVs and teams of UAVs in pick-and-place and car-landing applications in various environments that motivated the research presented in this paper. **a** object grasping in challenging desert conditions, **b** precise landing on a moving car, **c** object grasping in a green field, **d** UAVs cooperatively collecting objects in a desert environment <https://youtu.be/ns8n8xEMxD8>, **e** successful deployment of the system in the MBZIRC 2017 competition <http://mrs.felk.cvut.cz/mbzirc>, <https://youtu.be/ogmQSjkqp0>, **f** landing on a moving car during the competition with fine-tuned controller gains <https://youtu.be/zHs5LTLyBsc>

controller action is of prime importance, e.g. during the take-off and landing phases [10], and for a smooth touch-down phase. While grasping objects [25, 35, 39], the UAV is influenced by interference from the ground effect, and often by wind gusts. Manual tuning of one set of controller parameters takes dozens of trials and days of outdoor flying in a range of environment conditions (see examples of system deployment in various environments and flight conditions in Fig. 1). In addition, grasping any objects requires a new set of controller gains for various configurations of the gripping arms during transport tasks [41]. This is an even more significant problem for multi-UAV systems, where, in principle, each vehicle has to be tuned separately. This in fact inhibits the use of large teams of autonomous UAVs in tasks where interaction with the environment is required, for example, in the pick-and-place challenge, and in cooperative construction assembly. In fire fighting, the UAV model has to be changed due to release of the extinguishing medium [28]. These requirements have motivated the authors of the paper to introduce a family of methods for controller tuning. The aim is to achieve superior results (with prescribed tolerance) much more quickly that

with traditional approaches. This is a major contribution of our work, and it could significantly extend the application domain of UAVs. The proposed methods differ in algorithm picking intermediate search points, but can all be thought as different instances of the same general procedure, what is the main reason for presenting the family of methods in a single paper.

1.2 Related work

Today, the applications of UAVs [7] range from routing, data gathering, cellular communications, Internet of Things (IoT) networks, even to disaster management, to extend the capabilities of other techniques. All these cases require proper controller tuning to ensure good performance. Many model-based and model-free approaches to controller tuning have been developed. A review of 51 methods applicable for PIDs can be found in [31], and a range of tuning methods for fixed-parameter controllers are presented in [29].

As computational power has increased in recent years, a tendency has been observed to use meta-heuristics for the controller tuning problem. In the context of UAVs,

bio-inspired approaches have been attracting major attention. Since the beginning of the 21st century, at least 30 important new bio-inspired optimization methods have been developed. Several of the methods, accompanied by classical algorithms such as Particle Swarm Optimization (PSO), the Ant Colony Optimization, the Bat-inspired algorithm, the Artificial Bee Colony, etc., have been characterized, e.g. in [14] and in [16], from the viewpoint of application in UAVs. Paper [14] presents the results of research on PSO for a controller used in an unmanned rotorcraft pendulum and on multiple UAV formation control, and multiple UAV/UGV and heterogeneous coordinated control with the use of bio-inspired techniques.

Recent papers on tuning UAV controllers fall in three main categories: fuzzy logic applications to maintain self-tuning capabilities, adaptive gain scheduling approaches, and classical adaptive pole-placement techniques. References [1, 15, 36] present the adoption of fuzzy logic rules for tuning controller parameters. A basic disadvantage is that expert knowledge is needed for constructing the inference rules, preferably on the basis of a model. References [15, 32] adopt the gain scheduling approach, where a programme of gain changes is obtained that is derived for models for multiple operation conditions. Finally, in [12, 42, 44, 45], a model is used to determine appropriate controller gains in order to obtain nominal performance. As can be seen, all of the approaches inherently need a model. One can also make another division, namely the model is used to tune gains on the basis of some analytical calculations, or using other criteria, or by using machine learning techniques [38], with artificial neural networks. Such an approach is greatly time-consuming, whereas in our solution, one gets a rapid way of controller tuning in a reliable way, with deterministic time requirements, related to tolerance.

As stated in [17], on the basis of the Cuttlefish Optimization Algorithm, the new bio-inspired algorithms also have the potential to work as batch methods for tuning an altitude controller based on a UAV dynamical model. However, a model of the UAV is still required. In addition, all the methods mentioned above are still inapplicable in real-time conditions or using an on-board UAV computer. On-board, embedded systems are still too slow to allow successful in-flight optimization.

The short characteristics and grouping of approaches in the field has been given in Table 1.

1.3 Contribution and organization of the paper

In view of these drawbacks of methods that have been introduced here, our investigation concentrated on developing a simple and applicable method for tuning or fine-tuning the gains of standard fixed-parameter controllers, e.g., PD

or PID, for the UAV community, and on presenting them for tuning an altitude controller precisely. Fixed-parameter controllers are widely used in commercial and open source flight controllers (e.g. Pixhawk, Naze32, CC3D Open Pilot). However, due to the wide range of UAV applications, it is important to perform tuning in flight conditions, without needing to model the dynamics of the drone, using an iterative approach. To the best of the knowledge of the authors, the currently available commercial solution is for PID autotuning for roll, pitch and yaw angles (CleanFlight's G-Tune) for Naze/Multiwii platforms. Literature searches have not revealed any results for iterative autotuning altitude controllers that can be performed during flight, or the iterative learning is based on knowledge of a model, as in [37] or in [13].

Yet other tuning techniques may include hybridization approach including [3] fuzzy logic approach combined with MRAC MIT-oriented tuning, which inherently is limited by the MIT rule and its local convergence, or by building knowledge concerning the UAV on the basis of vision-gained information [2], to improve the performance of the system.

The AL-TUNE scheme presented in this paper performs automated tuning of controllers, using single-variable deterministic optimization methods for a given tolerance of the solution and the selected cost function, on the basis of an experimental data. The name AL-TUNE originates from the application, i.e., altitude controller tuning. For an initial discussion, see [19]. AL-TUNE consists zero-order branch-and-bound algorithms [34], modified and applied here to the tuning task (tuning a hexacopter's altitude controller with AL-TUNE in a real-world experiment, the RTK GPS system is used to evaluate the performance of the method). This new approach is very attractive, as it can be used in a range of on-flight tuning problems.

This paper presents a family of novel methods for real-time model-free optimization-based tuning of controllers, with experimental verification during flight to tune a PID-type altitude controller in a hexacopter UAV. The methods can be used to tune general controllers with a small number of parameters in an orderly manner, with gain bounds and with defined tolerance. This type of automated tuning is based on measurements only. It has a low computational cost, and it reduces the tuning time dramatically. The proposed tuning method needs no analytical model of the UAV, and takes no modeling issues into account. This makes it a simple and effective tool, unlike the methods available in the current literature. The possible absence of a model, combined with low computational complexity, rapid action and the advantages mentioned above, forms the novel contribution of the paper. An analysis of experimental records (ROS bags) obtained during preparations for the MBZIRC 2017 Treasure Hunt and Autonomous Landing

Table 1 Characteristics of approaches to controller tuning (G – general, Q – quadrotor, T – trirotor, F – fixed-wing UAV, PSO – particle swarm optimization, ACO – ant-colony optimization, NN – neural networks, FL – fuzzy logic)

Reference	UAV type	Tuning approach	Controller
[1, 15, 36, 42, 44]	Q	fuzzy logic	PID
[11]	G	NN	nonlinear
[5]	Q	ACO	pole-placement RST
[37]	Q	ILC	nonlinear
[13]	Q	Lyapunov-based MRAC	nonlinear
[33]	Q	model-based	PID
[4]	Q	MRAC/MIT	pole-placement RST
[7]	Q	MRAC	LQR
[38]	Q	NN	nonlinear
[14]	Q	PSO, ACO	PID
[16]	Q	PSO	FOPID
[17]	Q	PSO-like	PID et al
[26]	Q	various	nonlinear
[29, 31]	Q	various	PID
[3]	T	FL, MRAC	pole-placement RST
[2]	T	vision-based model	nonlinear
[12]	F	FL	nonlinear
[32]	F	FL	PI

challenges, where the precise altitude control was the crucial behaviour for success, have shown that manual tuning of a single set of control parameters for a single UAV required a mean time of ca. 45 minutes. In addition, the configuration parameters of the AL-TUNE scheme are easily-interpretable and provide the required performance of the optimization algorithm. This is a major advantage of the AL-TUNE scheme, enabling a streamlined configuration in various potential applications.

When prototypes of the algorithms were begin developed, a combination of the Robot Operating System (ROS) environment and the Gazebo environment was used to verify the zero-order methods. These initial Gazebo simulations are crucial for testing the correctness of the methods, for implementing, and for integrating them into a complex UAV system. However, in order to obtain real values, field experiments in real-world conditions are necessary.

The model of the UAV simulated in Gazebo reflects the dimensions, the specification and the parameter of a real UAV, together with all software solutions, the impact of the environment on the UAV, and fusion of the readings from sensors. This allowed extensive initial simulation tests to be performed in order to identify the strong and weak points of the methods, which were finally verified by means of real experiments.

The paper is structured as follows: Section 2 presents the software tools used to perform the initial tests on the methods are described, together with insights into the architecture of the UAV used to conduct experiments, in order finally to verify the applicability of the methods. Section 3 describes the novel on-line adaptation mechanism proposed

in the paper. Section 4 presents the family of optimization algorithms, and Section 5 refers to simulation results on a large data set, and the experimental results are presented in Section 6. Finally, conclusions and comments are presented in Section 7.

2 UAV Architecture and Software Tools

The position control system must be capable of ensuring good performance for large deviations from the hover configuration, or capable of compensating heavy wind. The proposed tuning scheme is used here to tune a pair of gains of a PID-type fixed-parameter altitude controller, with a small integral gain value, ensuring precise reduction of the steady-state position error. The proportional and derivative gains are tuned here to ensure superior performance in dynamical conditions, as these gains play a major role in obtaining good transient behaviour.

A nonlinear controller is used with the torque $\underline{m} \in \mathcal{R}^3$ and thrust force control $f_T \in \mathcal{R}$ as inputs, see [24] and [27]

$$\underline{m} = -k_R \underline{e}_R - k_\Omega \underline{e}_\Omega + \underline{\Omega} \times \underline{J} \underline{\Omega} - \underline{J} \left(\hat{\underline{\Omega}} \underline{R}^T \underline{R}_c \underline{\Omega}_c - \underline{R}^T \underline{R}_c \hat{\underline{\Omega}}_c \right), \quad (1)$$

$$f_T = - \left(-k_P \underline{e}_x - k_{ib} \underline{R} \int_0^t \underline{R}(\tau)^T \underline{e}_{xd} - \tau - k_{iw} \int_0^t \underline{e}_{xd} \tau - k_D \underline{e}_v - m g \underline{e}_3 + m \ddot{\underline{x}}_d \right) \underline{R} \underline{e}_3, \quad (2)$$

where $\mathbf{J} \in \mathbb{R}^{3 \times 3}$ is the inertia matrix with respect to the body frame, $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ is the orientation of the center of mass matrix, $\mathbf{R}_c \in \mathbb{R}^{3 \times 3}$ is the commanded value of the inertia matrix with respect to the body frame, $\underline{\Omega} \in \mathbb{R}^3$ is the angular velocity vector in the body fixed frame, $\underline{\Omega}_c \in \mathbb{R}^3$ is the commanded angular velocity vector in the body fixed frame, $\ddot{\mathbf{x}}_d$ is the desired acceleration, and k_P, k_D, k_R, k_Ω are positive definite terms (see [27]).

The e_x, e_v, e_R , and e_Ω denote, respectively, position, velocity, orientation, and angular rate errors, with the subscript $\cdot_c$ denoting the commanded value. As mentioned above, in our approach, the gains k_P and k_D are subject to tuning.

The experiments were conducted on a customized micro aerial vehicle platform, composed of DJI hexacopter frame F550 and E310 DJI motors, UAV PixHawk Autopilot, Real Time Kinematic GPS, the Rangefinder TeraRanger One sensor, used for obtaining altitude information when gathering data to obtain the cost function value during optimization, and an onboard Intel NUC-i7 PC (see Fig. 2). The system would also work with a standard GPS, as the GPS drift is slow, and the short-term position error is negligible when calculating the cost function values. In laboratory conditions, any motion capture system can be used in a similar way, e.g. VICON, OptiTrack, but the authors have focused on real-world applications. The proposed tuning scheme has negligible computational demands, as it is only necessary to collect a pair of cost function values and to pick two intermediate points by a simple inequality rule. This platform was developed for the Mohamed Bin Zayed International Robotic Challenge 2017, where its good performance was proven in the Treasure Hunt and in the Autonomous landing task on a moving car with a UAV [10, 39]. It should be stressed that such a powerful PC is not required, as stated above, for running the proposed methods. However, it was used for testing purposes, where it provided a detailed analysis of the results,

Fig. 2 The DJI F550 test platform

as a full ROS bag with complete sensory data can be processed and stored on-board in real time using this setup.

The position/altitude control system uses a mature MPC-based system for transforming a given trajectory primitive into a feasible trajectory primitive, ensuring a smooth transition along the reference. This model predictive control (MPC) approach modifies the reference trajectory by minimizing the cost function related to the fourth order derivative of the position, as the control inputs (torques) depend directly on it, see [8, 9]. The MPC scheme modifies the reference primitive which is to be tracked in our experiments by the altitude of the UAV. The modified reference primitive defines the desired dynamics of the UAV, and is indirectly related to the gains obtained from the AL-TUNE scheme. This is what motivated the authors use 3 different reference primitives in the initial simulation tests. The aim was to obtain a set of controller gains for the required dynamics of the control system.

For more information about the hardware that was used in the experiments see [35], with a detailed description of the hardware and software solutions, including path planning, low- and high-level control, etc. In addition, an extended description can be found in [35] of the implemented and tested precise multi-level estimator of the altitude of the UAV, which fuses data from GPS and from the laser Rangefinder sensor to enable smooth flight above the vehicle and landing on the vehicle.

3 Zero-Order Algorithms

3.1 Preliminaries

The family of optimization methods considered here allows tuning to be performed in cases when the model of a UAV is incomplete or undefined. The cost function for optimization purposes mirrors the performance quality of the control system, and is calculated iteration-by-iteration in a recursive fashion. Obviously, the global minimum of the cost function is unknown. It is therefore necessary to formulate the stopping criteria of the iterative tuning algorithm, with the use of which the optimized controller parameters are obtained. The algorithms are designed in such a way that the amount of time required to obtain final controller parameters of a defined quality is deterministic, and can be evaluated prior to the tuning process. Consecutive steps of the tuning algorithm require no information apart from altitude measurements at equally-spaced sampling periods. The computational complexity of the tuning algorithm is extremely low, due to the use of branch-and-bound algorithms, also termed zero-order algorithms, to find the solution.

The zero-order algorithms are based on finding the minimum of the cost function $f : \mathcal{R} \rightarrow \mathcal{R}$ inside the

feasible range of a parameter $[x^{(0-)}, x^{(0+)}]$ with $x^{(0+)} > x^{(0-)}$ of the (unimodal) cost function f . The argument of this function, in the one-dimensional case presented here, can be taken as the gain of the controller. If the cost function obtained by means of measurements/experiments is not unimodal, the algorithms stop at a local minimum.

Now, let a unimodal function f be given with the range $[x^{(0-)}, x^{(0+)}]$, and let us suppose it is obtained on the basis of measurements and results from some mathematical derivations, e.g., by measuring signals and incrementing some control-based cost function in the sampling periods. To find the minimum, two interior points for an evaluation of cost function f have to be taken inside the current range of a tuned parameter, in order to reduce this range in the next step. A symmetrical choice of the two interior points is proposed.

The algorithm for finding the minimum of unimodal one-argument cost function f can be summarized as follows:

- derive the number N of iterations necessary to reduce the considered range for a parameter of f for the chosen tolerance ϵ , related to the length of the initial range; assume that the iterative solution $\hat{x}^*$ satisfies $|x^* - \hat{x}^*| \leq \epsilon(x^{(0+)} - x^{(0-)})$, where $\hat{x}^*$ is in the center of $\mathcal{D}^{(N)}$, x^* is the actual (unknown) minimum, and superscripts $(0-)$ and $(0+)$ denote the left and right limits of the considered range in the 0-th iteration, respectively;
- for iterations $k = 1, \dots, N$:
 - 1) select two interior points $\hat{x}^{(k-)}$ and $\hat{x}^{(k+)}$ ($\hat{x}^{(k-)} < \hat{x}^{(k+)}$, $\{\hat{x}^{(k-)}, \hat{x}^{(k+)}\} \in \mathcal{D}^{(k-1)}$),
 - 2) reduce the range to $\mathcal{D}^{(k)}$, where:
 - a) $x^{(k+1)} \in \mathcal{D}^{(k)} = [x^{(k-1-)}, \hat{x}^{(k+)}]$ for $f(\hat{x}^{(k-)}) < f(\hat{x}^{(k+)})$,
 - b) $x^{(k+1)} \in \mathcal{D}^{(k)} = [\hat{x}^{(k-)}, x^{(k-1+)}]$ for $f(\hat{x}^{(k-)}) \geq f(\hat{x}^{(k+)})$,
 - 3) increase iteration counter $k := k + 1$,
- terminate the algorithm; the optimal iterative solution is $\hat{x}^* = \frac{1}{2}(x^{(N+)} + x^{(N-)})$.

When the number N of required iterations with the given tolerance ϵ is calculated, then for $k = 1, \dots, N$ it holds that:

$$\hat{x}^{(k-)} = x^{(k-1-)} + \rho_k \gamma_k, \quad (3)$$

$$\hat{x}^{(k+)} = x^{(k-1+)} + (1 - \rho_k) \gamma_k, \quad (4)$$

and ρ_k defines the contraction ratio, and γ_k defines the reference range (this pair of equations can be thought as a general view of a family of the proposed methods).

3.2 Proposed iterative zero-order in-flight tuning algorithms

The general rule for selecting a pair of new values for parameters (3) and (4) describes the family of algorithms that use different definitions for γ_k , and, as shown below, offer different performance in a noisy environment. In the paper, and during the experimental phase of the research, the following algorithms are considered (for a thorough description and for a comparison of their performance in computer tests, see [20, 21]):

- Fibonacci-search method (FIB),
- golden-search method (GLD),
- the equal division method (EQL),
- the dichotomy method (DIC) in two configurations.

Table 2 presents a summary of characteristic parameters taken by the zero-order methods, where $0 < \delta \ll 1$ is a small number, F_k denotes the k -th Fibonacci number, starting with $F_0 = 1$, $F_1 = 1$, and for FIB, GLD, and DIC it holds that $\gamma_k = x^{(k-1+)} - x^{(k-1-)}$, whereas for EQL the range by a constant factor related to the length of the initial range is reduced, i.e. $\gamma_k = \gamma_0$. It will be shown below that this difference gives rise to various contractions of the ranges along the iterations, making some of the methods more suitable in a real-world environment.

In the FIB, EQL and GLD methods, one of the points selected at the k -th iteration is known from the $k - 1$ -th iteration, due to the rule for selecting ρ_k . This can be used either to average the cost function values between iterations, e.g. to reduce the impact of external disturbances such as wind gusts on the current value of f in the k -th iteration, or to reduce the running time of the algorithm by skipping the current point, as the value of f has already been obtained.

The C-code, accompanied by their Matlab source files, is available under <https://github.com/AppliedControl>

Table 2 Summary of the specifications for the methods considered here

Method	Specification
FIB	$\rho_k = \frac{F_{N-k}}{F_{N-k+2}} (k = 0, 1, \dots, N-1),$
	$\rho_N = \frac{1}{2} - \delta,$
	$F_{N+1} \geq \epsilon^{-1}$
GLD	$\rho_k = \rho = \frac{3-\sqrt{5}}{2},$
	$(1 - \rho)^N \leq \epsilon$
EQL	$\rho_k = \epsilon \quad (\epsilon^{-1} \in \mathcal{N}, k = 0, 1, \dots, N-1),$
	$\rho_N = \epsilon - \delta,$
	$N \geq \epsilon^{-1} - 1$
DIC	$\rho_k = \frac{1}{2} - \delta,$
	$\frac{1}{2^N}(1 - \delta) + \delta \leq \epsilon$

[Techniques/AL-Tune](#) under GNU GPL license. Please also visit act.put.poznan.pl for further information on the topic.

4 AL-TUNE Scheme for Tuning of Two-Parameter Altitude Controller

4.1 Presentation of the Scheme

The final optimized values for the gains of a two-parameter controller can be found easily on the basis of the methods presented in the previous section. The running time of the tuning procedure is a function of the tolerance ϵ , expressed as the relative length (i.e., with reference to the initial range) of the final range within which the actually final, though unknown, gain is, and the number of bootstrap cycles N_b defined by the user. In the application considered here, the bootstrap cycle is composed by solving two consecutive optimization tasks on the basis of measurements (incremental calculation of f). One of the parameters is changed at a time, keeping the remaining parameter constant, either with the value initially stipulated by the user, or with the value resulting from the previous bootstrap. In this way, the first parameter is tuned first, keeping the second one constant, and then the second parameter is tuned, keeping the first one constant.

Of course, the proposed AL-TUNE scheme can be extended to a greater number of parameters, which results in longer convergence of the iterative minimizer. A good solution for controllers with a greater number of parameters is either to split the tuning into stages, where independently working controllers, e.g. in angle and rotational velocity control loops, are tuned separately. Another effective approach is to fix the gain of the controller, tuning the remaining gains, and to repeat the experiments for a discrete set of the gain that has been defined a priori. The behaviour of the proposed algorithms can be interpreted from the viewpoint of the steepest-descent method [20], as the local minimizer obtained in the termination phase of the bootstrap is used to start the next iteration.

In a two-parameter framework, the vector of parameters $\underline{x} = [x_1, x_2]^T$ of the cost function $f(\underline{x})$ related to the performance of the control system can define the gains of the controller. From the tuning viewpoint, the cost function reflects the degree to which the gains are tuned, and increases for mistuned gains. The following assumptions are made:

- admissible ranges of $\underline{x}$, $\mathcal{D}_i^{(0)} = [x_i^{(0-)}, x_i^{(0+)}]$, $i = 1, 2$, are known, and are usually related to the ranges of safe operation of the UAV (these come from initial tests carried out with the UAV or as a span of the values of the

controller gains, including the default controller gains, in their vicinity),

- a zero-order method is used to find $\hat{\underline{x}}^*$, and is based on bootstraps and algorithms from Section 3,
- the cost function f may be related to a measure of the performance of the control system; here it has been assumed to measure the sums of the absolute tracking error samples in a horizon N_{ref} of the selected number of samples, related to the period of the reference primitive r_{alt} . The 1-norm tracking error function has been used since the measured errors given in meters are relatively small. This, in addition, shows the great robustness of methods that identify the minimizer on the basis of noisy measurements, and imposes a linear penalty on increasing error amplitudes. In this way, the cost function, which is in fact the performance index, is incremented in each sampling period T_s as a linear function of the current absolute value of the tracking error,

$$f = \sum_{i=1}^{N_{\text{ref}}} |r_{\text{alt}}(iT_s) - y_{\text{alt}}(iT_s)|. \quad (5)$$

- for a given tolerance ϵ , the number N of necessary iterations can be calculated, and the number $2N$ denotes the number of calculations of the cost function for the combination of controller parameters in a single bootstrap (N iterations are necessary for each parameter),
- the stopping criterion is expressed by the number of bootstraps N_b chosen by the user. This reflects the time needed for completing the tuning, which can, in turn, be bounded by the voltage drop in the UAV's batteries.

The optimized-tuning algorithm can be given in a general form (see also the flowchart in Fig. 3, for an explanation of actions taken in a single bootstrap, and in Fig. 4 for an explanation of the actions taken in a complete tuning algorithm):

- 0) put the bootstrap cycle counter $l = 0$; for given $\mathcal{D}_i^{(l)}$ ($i = 1, 2$) define ϵ , N_b , and the initial value of the second parameter $x_2^{(l)}$ (take $\hat{x}_2^{(l)*} = x_2^{(l)}$), put $l := l + 1$,
- 1) find the optimal $\hat{x}_1^{(l)*}$, using any of the zero-order methods, with the second parameter equal to $\hat{x}_2^{(l-1)*}$, i.e., taken from the prior bootstrap sequence,
- 2) find the optimal $\hat{x}_2^{(l)*}$ using the same method as in Step 1, with the first parameter equal to $\hat{x}_1^{(l)*}$,
- 3) if $l < N_b$, increase the bootstrap cycle counter $l := l + 1$, and proceed to Step 1, otherwise stop the algorithm – the optimal solution $\hat{\underline{x}}^* = [\hat{x}_1^{(l)*}, \hat{x}_2^{(l)*}]^T$ has been obtained after N_b bootstrap cycles, as desired.

Fig. 3 General flowchart of a bootstrap sequence within AL-TUNE

In each of the methods, the cost function has been calculated as presented above, processing the values of the low-pass filtered altitude tracking error in the defined time horizon of 50 samples, with a sampling period of 0.05 sec. After the performance index (cost function) has been gathered, the optimization step is carried out either to change the parameters, or to reduce the range for a parameter, followed by several samples, here: 10, allowing transients to decay whenever the controller parameters are changed.

At each step of the zero-order algorithm, two interior points in the range considered in the prior iterations are taken, related to two different controller gain values. After the values of the cost function have been gathered for a pair of points, a decision can be made on how to reduce the range. For all the methods with the exception of DIC, one of the points starting from the second iteration has already been evaluated. On this basis, as stated in the previous Section, the algorithm can be further modified either to take

the value of the cost function as the mean value of all prior cost function values in this configuration of gains, or this combination of gains can be skipped.

4.2 Discussion of Optimality

First, let $f(x_1, x_2)$ be a unimodal function within admissible ranges $\mathcal{D}_i^{(0)}$ ($i = 1, 2$). Obviously, for arbitrary $a \in \mathcal{D}_1^{(0)}$ and $b \in \mathcal{D}_2^{(0)}$ the following one-argument functions $f(x_1, b)$ and $f(a, x_2)$ are, respectively, unimodal. At any stage of a bootstrap sequence, the minimized function is treated as a one-argument function, denoted henceforth as $g(x)$, where, without loss of generality, $x = x_1$ or $x = x_2$. Since $g(x)$ has a unique minimizer

$$x^* = \arg \min_{\alpha \leq x \leq \beta} g(x) \quad (6)$$

within the range $[\alpha, \beta]$ ($\alpha < \beta$), the following $\alpha \leq x^{(1)} < x^{(2)} \leq x^*$ implies $g(x^{(1)}) > g(x^{(2)})$, and $x^* \leq x^{(1)} < x^{(2)} \leq \beta$ implies $g(x^{(1)}) < g(x^{(2)})$. It is assumed that $g(x)$ is calculated/measured with finite accuracy. In this one-dimensional search, g is evaluated at N distinct points $x^{(1)}, x^{(2)}, \dots, x^{(N)}$, where the placement of $x^{(k+1)}$ is a function of $g(x^{(j)})$ for $j \leq k$. The search procedure terminates when successive reduction of the initial interval is achieved.

Let us assume that k function evaluations have been performed, and

$$\hat{x} = \arg \min_{1 \leq i \leq k} \left(g(x^{(i)}) \right), \quad (7)$$

$$\mathcal{D}_L^{(k)} = \left\{ x^{(j)} : x^{(j)} < \hat{x}, j \leq k \right\} \cup \{\alpha\}, \quad (8)$$

$$\mathcal{D}_R^{(k)} = \left\{ x^{(j)} : x^{(j)} > \hat{x}, j \leq k \right\} \cup \{\beta\}, \quad (9)$$

$$x_L^{(k)} = \inf \mathcal{D}_L^{(k)}, \quad x_R^{(k)} = \sup \mathcal{D}_R^{(k)}. \quad (10)$$

Now, as $g(x)$ is unimodal, the true minimizer x^* must obviously lie within $[x_L^{(k)}, x_R^{(k)}]$, which also holds after N iterations.

Performing the same procedure in the second stage of any bootstrap sequence yields an improved estimate of the optimal point in a 2D search space, according to the unimodality assumption of the function. The procedure stops when the rectangular range has relative dimensions below ϵ .

It is now obvious that for the proposed method for a unimodal function $f(x_1, x_2)$, the bootstrap sequences result in the steepest descent-like behaviour with the search direction in the k -th iteration orthogonal to the direction from the $k - 1$ -th direction in each bootstrap. In the case of multimodal f , the function g is multimodal too, and the obtained minimizer $\hat{x}^{(N)}$ may be local, resulting in the minimizer after N iterations in the following state of the bootstrap sequence again being local, with an improved cost function value with respect to the previous bootstrap stage, which simply results from the decision that is taken when

Fig. 4 Simplified flowchart of AL-TUNE

selecting a new pair of points. This is a common feature of all single-pass optimization algorithms, being the cost of a very low computational complexity connected with: gathering of samples to calculate the performance index, and comparing its values across the iterations only.

5 Simulations with an Accurate Model

In order to evaluate the proposed methods, an extensive series of simulations was performed. The sum of absolute tracking errors performance index (here: cost function $f(k_P, k_D)$) was averaged in a grid of 24×24 combinations of two controller parameters from the safe range. This data was gathered in a long time horizon, averaged over 20 periods of the reference signal for each of the 576 combinations. In this way, the surface of the performance index was obtained, presented in Fig. 5a–e in a logarithmic scale. The choice of the scale followed from a very flat graph of the performance index surface, which would be very problematic for other non-zero-order algorithms.

Having collected the values of the performance indexes, three series of tuning experiments were conducted for fast (F, $v_{\max} = 10 \frac{\text{m}}{\text{sec}}$, $a_{\max} = 100 \frac{\text{m}}{\text{sec}^2}$), medium (M, $v_{\max} = 3.33 \frac{\text{m}}{\text{sec}}$, $a_{\max} = 11.11 \frac{\text{m}}{\text{sec}^2}$), and slow (S, $v_{\max} = 1.67 \frac{\text{m}}{\text{sec}}$, $a_{\max} = 2.78 \frac{\text{m}}{\text{sec}^2}$) reference primitives in the form of low-pass filtered square waves with a period of 12 sec, depicted in Fig. 5a–e in green, blue and red, respectively.

During all simulations, it was assumed that:

- 2 bootstraps are performed,
- k_P is sought in the range [4, 12], and k_D in [2, 7],
- $\epsilon = 0.05$ in DIC1, GLD, FIB, or $\epsilon = 0.125$ in EQL, DIC2,
- $\delta = 0.0625$ in EQL, $\delta = 0.01$ in FIB, $\delta = 0.04$ in DIC1, and $\delta = 0.1$ in DIC2,
- a total of 48 iterations are required in FIB, DIC2 and GLD (6 iterations per half of a bootstrap), while 56 iterations are required in DIC1, EQL (7 iterations),
- the initial values for the second parameter (k_D) take the following values: 1, 3, 5, 7, 9,
- for all methods apart from the DIC family, the algorithms used the calculated/averaged cost function, or skipped the points known from previous iterations, which gave rise to 45 simulations/algorithm, and 15 for DICs.

As can be seen, all the methods find the local minimum but differ in their ability to reproduce the result in another simulation. This results from the relative difference between the ends of the ranges along consecutive iterations. Figure 10 shows that the EQL method preserves a large difference between two configurations of the controller parameters through all iterations. This enables the algorithm

to gather the cost function in sufficiently varying work conditions, supporting a good decision on how to bound the new range. In DIC1, however, the range decreases rapidly, which causes the controller parameters to be found almost randomly (see Fig. 5d). When the difference between the ends of a range is moderate (see DIC2 or FIB), the tuned parameters are much closer to the true minimum point which is especially important in experiments. When the difference is small, and external disturbances affect the measurements, gathering performance indices for two gains that are too close together may result in performance deterioration of the tuning method, making the gains the same from the performance viewpoint. This was the main motivation for validating the performance of the DIC method in two configurations.

If we were to consider altering the tolerance ϵ , the resulting change in the required number of iterations N is depicted in Fig. 11. As can be seen, N grows rapidly for the EQL and DIC2 methods, which proved to have the best performance here. This is also strongly related to Fig. 10, where the methods for ensuring a fair distance between the selected gain pairs use measurements sufficiently different from each other, in order to make the pairs distinguishable. This is the contrast with the results presented in Fig. 11, where the methods require a greater number of iterations to be performed.

In addition, a series of 10 tuning simulations was conducted with the UAV and EQL algorithm in two configurations, namely: with a cylindrical load of 0.5 kg, diameter 0.3 m, height 0.1 m attached to the UAV, and without the load. In the first case, the gains averaged over 10 simulations were: $k_P = 10.8$, $k_D = 5.16$, whereas in the second case, $k_P = 9$, $k_D = 3.72$. To show the impact of proper tuning on the behaviour of a UAV, 4 long-horizon simulations were conducted with 100 repetitions of the reference primitive, as summarized in Table 3.

6 AL-TUNE: Experimental Results

The proposed methods were verified using experiments carried out on the same UAV (Fig. 6). The cost function was built as the sum of the absolute tracking error samples, the ranges of the parameters, and all other parameters were the same as for the simulations. A low-pass filtered tracking error was used with the M trajectory primitive with 12 sec length, changing in the range $2 \div 3$ m. Each experiment had the following stages: arm the UAV, reach the desired position and altitude of 3 m, tune, and land. The experiments were conducted in fully autonomous mode, with the ground true taken from the differential GPS. The results presented here come from the tuning stage. In all the cases, the initial value of k_D was set to 9.

Fig. 5 Tuning results for a family of methods from 3 series of tuning experiments and various initial values of k_D (see the body of the text for further information), for a cost function with a flat graph near the local optimum point

Table 3 Std. deviation of the tracking error in four load-to-gains configurations (A – attached, B – detached load)

	Load configuration A	Load configuration B
gains A	0.0229	0.0187
gains B	0.0350	0.0171

The experiments were conducted in windy conditions. Variable wind gusts are the most problematic environmental parameter for UAV stability. The experiments and also numerous other deployments of the system in various environmental conditions [30, 40, 43] have shown that the selected SO3 controller (1), and (2) with properly-tuned parameters provides sufficient stability.

Fig. 6 Caption from the hexacopter altitude controller tuning with AL-TUNE, in Gazebo (upper row), and in the experiments (lower row)

The rates of pitch, roll and yaw, and also the altitude, are controlled by the PixHawk embedded stabilizer. The special orthogonal group SO3 controller described briefly in Eqs. 1, and 2 was implemented on the NUC-i7 PC on-board computer, and supplies PixHawk (PX4), using internally PID controllers, with the altitude reference. The controllers of all the axes are fully-decoupled. The control inputs, i.e., the rates, for the altitude rate controller are orthogonal and do not influence each other. The outer-loop controller based on SO3 ensures control in position, velocity and acceleration, and its gains are tuned here by AL-TUNE.

Figure 7 presents the controller gains during tuning. As can be seen, FIB, GLD and DIC2 terminate earlier, whereas EQL and DIC1 terminate after 672 sec. The differences discussed in the section dealing with simulations can also be observed here. The final gains are approx. in the same area, as the minimum cost function area from Fig. 5a–e.

Figure 8a presents the tracking errors. It can be seen that, due to proper tuning, the absolute tracking error converges to zero. The large tracking error values are due to wind gusts, which did not cause the tuning to go wrong. Figure 8b, and c present the difference in tracking quality

Fig. 7 Tuning of: **a** k_P , **b** k_D . For a video, see: <https://youtu.be/Njv7xVGeKks>

Fig. 8 **a** tracking errors e for all considered tuning methods, **b** mistuned gains (DIC1) resulting in poor performance, **c** well-tuned gains (DIC1) resulting in improved performance

Fig. 9 **a** pitch θ ; **b** roll ϕ ; **c** yaw ψ angles in ($^\circ$) for EQL tuning case

Table 4 Comparison of the results of experiments (1st row) and a statistical overview of the simulations (other rows), obtained for reference primitives M and cost functions with very wide local minimum zones, resulting in gains yielding in a highly comparable performance

Method	FIB ($\epsilon = 0.05$)		EQL ($\epsilon = 0.125$)		GLD ($\epsilon = 0.05$)		DIC1 ($\epsilon = 0.05$)		DIC2 ($\epsilon = 0.125$)	
	k_P	k_D	k_P	k_D	k_P	k_D	k_P	k_D	k_P	k_D
experiment	9.57	4.26	9.98	3.31	8.81	4.31	8.05	3.44	6.36	2.99
mean (sim)	9.55	3.38	9.64	3.49	8.86	3.48	8.67	3.25	8.40	4.05
std dev	1.41	1.11	0.45	0.34	1.79	1.03	1.64	0.67	1.16	0.73
min value	5.76	2.12	8.70	2.98	5.92	2.09	7.52	2.65	7.46	2.84
max value	11.38	5.69	10.42	3.94	11.69	5.91	11.54	4.20	10.25	4.73

Fig. 10 Relative difference between the ends of the ranges across iterations

on the basis of the DIC1 results for poorly-tuned and well-tuned parameters that make a difference in altitude control, in order to depict the need for good tuning of the controller parameters, adapting the gains to the mission requirements (i.e., the required speed of the transients). For example, DIC1 can be generalized to other methods from the family. In order to depict the behaviour of the UAV during the tuning procedure, a sample case for EQL tuning has been presented in Fig. 9, to show changes in roll, pitch, and yaw angles. Slight changes in the angles are given rise by environmental disturbances, such as wind gusts, and do not impede the tuning process at large.

Table 4 gives a statistical comparison of simulation vs. experimental results, presenting the results of a single experimental run for each method, and the series of simulations from Figs. 5a–e. As can be seen, the best results are obtained by the FIB and EQL methods, which yield controller gains close to the experimental results and the simulation results. However, the DIC and GLD methods (see Fig. 10) offer contraction ratios that provide indistinguishable gains in the initial iterations, making the tuning process a random walk. It therefore cannot be applied in the cases discussed here, i.e. when consecutive values of the cost function differ only very slightly (Fig. 11).

As a result of the extensive tests of the AL-TUNE scheme, it has been found that the best results are obtained by methods that preserve the widest possible ranges for the sought parameters during all iterations (see Fig. 10),

and that at the same time have the least standard deviation between the final points obtained from different simulation runs, namely EQL, and secondly DIC2. Very wide ranges for the parameters make the controller gains selected between the iterations distinguishable, despite any disturbances, wind gusts or noisy altitude measurements, and reduce their impact on the performance of the methods.

7 Conclusions

This paper has presented a family of new real-time autotuning methods for controllers based on a zero-order optimization algorithm. The following advantages of the method can be listed on the basis of the experiments reported here:

- for defined feasible ranges, the methods can be used for optimization-based tuning of controllers,
- a suitable method can be selected, depending on the level of the external forces acting on a UAV (e.g. wind gusts), by changing the controller gains during the experiment sufficiently to enable a comparison of the cost function, to carry on the optimization; in the end, the tuning method considerably improves the tracking performance,
- the proposed methods are fast, and do not require huge memory. In addition, they are very versatile and can be used to tune low-level controllers and also high-level controllers,

Fig. 11 Minimum number of iterations N vs. tolerance ϵ

- a single bootstrap duration time can be greatly reduced for fast-dynamics UAVs, or by omitting gain configurations that have already been checked in prior iterations in the tuning algorithm,
- for well-defined initial ranges, the proposed method results in a search for the optimum in the domain of all stabilizing controllers. This is because the predefined ranges define, in general, the interior of the closed-loop stability region. The method therefore operates safely, eliminating potential threats for operators.

The proposed approach enables rapid tuning, and unlike previously-mentioned tuning techniques, requires virtually no prior knowledge from a model of a UAV. The versatility of the family of tuning methods can be used to tune controller parameters from scratch, to adapt them for various work regimes for various loads, and even for fine-tuning existing controllers, pushing them to their limits performance limits.

8 Future Work

When a controller working in a control system has already been tuned (e.g. with one of the methods presented in the introduction), and the dynamic properties of the plant change, because loads are gripped or are released, because operating conditions change, or because the model being taken into account results in poor control quality, the proposed approach can find new superior controller parameters, e.g., in the vicinity of gains used in prior, by defining ranges to include the prior set of admissible parameters. It can potentially be used in various tasks, especially for applying tuning in aerial manipulation, for grasping, and for perching, when the change in the center of mass has a radical impact on control performance, what could be an interesting area for future research.

The introduced family of novel tuning methods is planned to be extended to the case of using a pair of UAVs, what has been already done to some extent, see [18], to further reduce the running time of the algorithm, as well as to increase the resilience of the method against environmental disturbances. Secondly, the methods should be considered with respect to increase the efficiency of the UAV deployment in a last-centimeter delivery systems operating in the vicinity of humans, see [23], or to increase the precision of cooperation of swarms of UAVs (as MBZIRC 2023-related issues, see [22]).

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10846-021-01441-y>.

Acknowledgements The presented work has been supported by Poznan University of Technology under grant No. 214/SBAD/0220,

as well as by the Czech Science Foundation (GAČR) under research project No. 20-10280S, EU H2020 project AERIAL CORE, No. 871479 and CTU grant No. SGS20/174/OHK3/3T/13.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Abbasi1, E., Mahjoob, M.J., Yazdanpanah, R.: Controlling of Quadrotor UAV Using a Fuzzy System for Tuning the PID Gains in Hovering Mode. Taipei, Proceedings of the SICE Annual Conference (2010)
2. Aguilar, W.G., Salcedo, V.S., Sandoval, D.S.: Cobena B Development of a Video-Based Model for UAV Autonomous Navigation. In: Barone, D., Teles, E., Brackmann, C. (eds.) Computational Neuroscience. LAWCN 2017. Communications in Computer and Information Science, vol. 720, pp. 94–105. Springer, Cham (2017)
3. Ali, Z.A., Wang, D., Aamir, M.: Fuzzy-Based Hybrid control algorithm for the stabilization of a Tri-Rotor UAV. *Sensors* **16**, 652 (2016)
4. Ali, Z.A., Li, X.: Controlling of an Under-Actuated quadrotor UAV equipped with a manipulator. *IEEE Access* **8**, 34664–34674 (2020). <https://doi.org/10.1109/ACCESS.2020.2974581>
5. Ali, Z.A., Zhangang, H., Wang, B.: Cooperative Path Planning of Multiple UAVs by using Max–Min Ant Colony Optimization along with Cauchy Mutant Operator. *Fluct. Noise Lett.* **20**(01), 2150002 (2021). <https://doi.org/10.1142/S0219477521500024>
6. Ali, Z.A., Li, X.: Modeling and controlling of quadrotor aerial vehicle equipped with a gripper. *Measur. Control* **52**(5–6), 577–587 (2019). <https://doi.org/10.1177/0020294019834040>
7. Alzahrani, B., Oubbati, O.S., Barnawi, A., Atiquzzaman, M., Alghazzawi, D.: UAV Assistance paradigm: State-of-the-art in applications and challenges. *J. Netw. Comput. Appl.* **166**(102706) (2020)
8. Báča, T., Loianno, G., Saska, M.: Embedded Model Predictive Control of Unmanned Micro Aerial Vehicles. In: 2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR), pp. 992–997. Międzyzdroje, Poland (2016)
9. Báča, T., Stepan, P., Saska, M.: Autonomous landing on a moving car with unmanned aerial vehicle. The European Conference on Mobile Robotics (ECMR), pp. 294–299 (2017)
10. Báča, T., Štěpán, P., Spurný, V., Saska, M., Pěnička, R., Loianno, G., Thomas, J., Kumar, V.: Autonomous Landing on a Moving Vehicle with Unmanned Aerial Vehicle. *Journal of Field Robotics* (2017)

11. Chen, F.C.: Back-propagation neural networks for nonlinear self-tuning adaptive control. *IEEE Control Syst. Mag.* **10**(3), 44–48 (1990)
12. Di, L., Chao, H., Han, J., Chen, Y.Q.: Cognitive multi-UAV formation flight: principle, low-cost UAV testbed, controller tuning and experiments, *Proceedings of the ASME 2011 International Design Engineering Technical Conferences Computers and Information in Engineering Conference*, pp. 917–927, Washington (2011)
13. Diao, C., Xian, B., Yin, Q., Zeng, W., Li, H., Yang, Y.: A nonlinear adaptive control approach for quadrotor UAVs. In: *Proceedings of the Asian control conference (ASCC)*, pp. 223–228 (2011)
14. Duan, H., Li, P.: *Bio-Inspired Computation in Unmanned Aerial Vehicles*. Springer, Berlin (2014)
15. Gautam, D., Ha, C.: Control of a Quadrotor Using a Smart Self-Tuning Fuzzy PID Controller. *Int. J. Adv. Robot. Syst.* **10**(1), 380:2013 (2013)
16. Giernacki, W., Coelho, J.P.: Evolutionary-based tuning approach of $PI^{\lambda}D^{\mu}$ fractional-order speed controller for multirotor UAVs, *13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO)*, pp. 312–317 (2018)
17. Giernacki, W., Espinoza-Fraire, T., Kozierski, P.: Cuttlefish optimization algorithm in autotuning of altitude controller of unmanned aerial vehicle (UAV), *advances in intelligent systems and computing*, pp. 841–852. Springer (2018)
18. Giernacki, W., Horla, W.M.: In-flight Efficient Controller Auto-tuning using a Pair of UAVs *Saska IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1300–1307, Las Vegas (2020)
19. Giernacki, W., Horla, D., Báča, T., Saska, M.: Real-time model-free minimum-seeking autotuning method for unmanned aerial vehicle controllers based on Fibonacci-search algorithm. *Sensors* **19**, 312 (2019)
20. Horla, D.: *Computational methods in optimization (in Polish)*, 2nd ed, pp. 358. Publishing House of Poznan University of Technology (2016)
21. Horla, D.: Performance evaluation of iterative methods to unconstrained single variable minimization problems. *Stud. Autom. Inf. Technol.* **38**, 7–34 (2013)
22. Horla, D., Giernacki, W., Cieslak, J., Campoy, P.: Altitude Measurement-Based Optimization of the Landing Process of UAVs. *Sensors* **21**, 1151 (2021)
23. Kornatowski, P.M., Bhaskaran, A., Heitz, G.M., Mintchev, S., Floreano, D.: Last-Centimeter Personal Drone Delivery: Field Deployment and User Interaction. *IEEE Robot. Autom. Lett.* **3**(4), 3813–3820 (2018)
24. Lee, T., Leok, M., McClamroch, N.H.: Nonlinear robust tracking control of a quadrotor UAV on SE(3). *Asian J. Control* **15**(2), 391–408 (2013)
25. Loianno, G., Spurný, V., Baca, T., Thomas, J., Thakur, D., Krajník, T., Zhou, A., Cho, A., Saska, M., Kumar, V.: Localization, Grasping and Transportation of Magnetic Objects by a team of MAVs in Challenging Desert like Environments. *IEEE Robot. Autom. Lett.* **3**(3), 1576–1583 (2018)
26. Mahony, R., Kumar, V., Corke, P.: Multirotor aerial vehicles: modeling, Estimation, and Control of Quadrotor. *IEEE Robot. Autom. Mag.* **19**(3), 20–32 (2012)
27. Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2520–2525 (2011)
28. Merino, L., Martínez-de Dios, J.R., Ollero, A.: Cooperative unmanned aerial systems for fire detection, monitoring, and extinguishing. In: Valavanis, K.P., Vachtsevanos, G.J. (eds.) *Handbook of unmanned aerial vehicles*, pp. 2693–2722. Springer, Netherlands (2015)
29. O'Dwyer, A. *Handbook of PI and PID Controller Tuning Rules*, 3rd edn. Imperial College Press, London (2009)
30. Pěnička, R., Faigl, J., Saska, M.: Variable Neighborhood Search for the Set Orienteering Problem and its application to other Orienteering Problem variants. *Eur. J. Oper. Res.* **276**(3), 816–825 (2019)
31. Panda, R.: *Introduction to PID Controllers – Theory, Tuning and Applications to Frontier Areas*. In-Tech (2012)
32. Poksawat, P., Wang, L., Mohamed, A.: Gain scheduled attitude control of Fixed-Wing UAV with automatic controller tuning. *IEEE Trans. Control Syst. Technol.* **26**(4), 1192–1203 (2018)
33. Pounds, P., Bersak, D.R., Dollar, A.M.: Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control. *Auton. Robot.* **33**(1), 129–142 (2012)
34. Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: a review of algorithms and comparison of software implementations. *J. Glob. Optim.* **56**, 1247–1293 (2013)
35. Saska, M., Baca, T., Spurný, V., Loianno, G., Thomas, J., Krajník, T., Stepan, P., Kumar, V.: Vision-based high-speed autonomous landing and cooperative objects grasping - towards the MBZIRC competition. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems – Vision-based High Speed Autonomous Navigation of UAVs (Workshop)*, pp. 1–5, Daejeon (2016)
36. Sangyarn, T., Laohapiengsak, P., Chongcharoen, W., Chen, I.N.: Path tracking of UAV using self-tuning PID controller based on fuzzy logic. *Proceedings of SICE Annual Conference*, pp. 1265–1269, Taipei (2010)
37. Schoellig, A.P., Mueller, R.L., D'Andrea, R.: Optimization-based iterative learning for precise quadcopter trajectory tracking. *Auton. Robot.* **33**(1–2), 103–127 (2012)
38. Song, Y., Naji, S., Kaufmann, E., Loquercio, A., Scaramuzza, D.: *Flightmare: A Flexible Quadrotor Simulator*. *Conference on Robot Learning* (2020)
39. Spurný, V., Báča, T., Saska, M., Pěnička, R., Krajník, T., Loianno, G., Thomas, J., Thakur, D., Kumar, V.: Cooperative Autonomous Search, Grasping and Delivering in Treasure Hunt Scenario by a Team of UAVs. *J. Field Robot.* **36**(1), 125–148 (2018)
40. Stepan, P., Krajník, T., Petrlik, M., Saska, M.: Vision techniques for on-board detection, following and mapping of moving targets. *J. Field Robot.* **36**(1), 252–269 (2019)
41. Suarez, A., Soria, P.R., Heredia, G., Arrue, B.C., Ollero, A.: Anthropomorphic, Compliant and Lightweight Dual Arm System for Aerial Manipulation. In: *Intelligent Robots and Systems (IROS)*, 2017 IEEE/RSJ International Conference on, pp. 992–997, Vancouver (2017)
42. Vianna Junior, J.C., De Paula, J.C., Leandro, G.V., Bonfim, M.C.: Stability Control of a Quad-rotor Using a PID Controller. *Braz. J. Instrum. Control* **1**(15), 15–20 (2013)
43. Walter, V., Staub, B., Saska, M.: UVDAR System For Visual Relative Localization with application to Leader-Follower Formations of Multirotor UAVs. *IEEE Robot. Autom. Lett.* **4**(3), 263–2644 (2019)
44. Yang, J., Cai, Z., Lin, Q., Wang, Y.: Self-tuning PID Control Design for Quadrotor UAV Based on Adaptive Pole Placement Control. *Chinese Automation Congress*, Changsha (2013)
45. Zemalache, K.M., Beji, L., Maaref, H.: Control of a Drone: Study and Analysis of the Robustness. *J. Autom. Mob. Robot. Intell. Syst.* **2**(1), 33–42 (2008)

Dariusz Horla (Member, IEEE) received the M.Sc. degree from Poznan University of Technology, in 2002, and the Ph.D. degree in 2005, and D.Sc. degree in 2013 from the same University, respectively. Since 2020 he is an associate professor at Poznan University of Technology, where he has founded and since leads the Applied Control Techniques research group. He was a Visiting Scholar with the University of Madrid, FCT Nova in Lisbon, and Czech Technical University in Prague. He is the author or a coauthor of more than 60 publications in peer-reviewed conferences, more than 20 publications in impacted journals and more than 40 papers in the other journals. He focuses on using control or optimization techniques, optimal, adaptive or robust control methods, developing linear matrix inequality conditions for anti-windup compensation with no directional change in controls, and tuning problems. He was a member of UPM-UPO-PUT team in the MBZIRC 2020 robotic competitions in Abu Dhabi in UAE where they scored the third place in the Grand Challenge.

Wojciech Giernacki (Member, IEEE) received the M.Sc. degree in control engineering and robotics from the Poznan University of Technology in 2006 and Ph.D. degree in 2011. Since 2021, he is an associate professor at the Institute of Robotics and Machine Intelligence of the same university. He founded and heads the PUT AeroLab and Unmanned Aerial Vehicles Research Group (<http://uav.put.poznan.pl/>). He was lecturing at the Shanghai University, Technical University of Madrid, Czech Technical University in Prague and Technical Institute in Braganca. He is an author or co-author of more than 70 publications in peer-reviewed international conferences, articles in impacted journals and 1 book. He serves as a reviewer for several impacted journals as well as a program committee member of relevant international conferences. His scientific interests are focused around the issues of UAVs, especially robust and adaptive control, optimization techniques, as well as data fusion from sensors. He was a member of UPM-UPO-PUT team in the MBZIRC 2020 robotic competitions in Abu Dhabi where they scored the third place in the Grand Challenge.

Tomas Báča (Graduate Student Member, IEEE) received the M.Sc. degree from Czech Technical University in Prague, Czech Republic, where he is currently pursuing the Ph.D. degree in distributed remote sensing. Since 2014, he has been a part of the Multi-Robot Systems Laboratory with CTU Prague. He focuses on distributed radiation sensing with unmanned aerial vehicles and planning and control for unmanned aerial vehicles. He was a Key Member of CTU-UPenn-UoL and CTUUPENN-NYU teams in the MBZIRC 2017 and 2020 robotic competitions in Abu Dhabi.

Vojtech Spurny received the M.Sc. degree from Czech Technical University in Prague, Czech Republic, where he is currently pursuing the Ph.D. degree in methods of planning and coordination for unmanned aerial vehicles. Since 2014, he has been a part of the Multi-Robot Systems Laboratory with CTU Prague. He was a Key Member of CTUUPenn-UoL and CTUUPENN-NYU teams in the MBZIRC 2017 and 2020 robotic competitions in Abu Dhabi, United Arab Emirates.

Martin Saska (Member, IEEE) received the M.Sc. degree from Czech Technical University in Prague, in 2005, and the Ph.D. degree from the University of Wuerzburg, Germany. Since 2009, he has been a Research Fellow with Czech Technical University in Prague, where he has founded and since leads the Multi-Robot Systems Laboratory and co-founded the Center for Robotics and Autonomous Systems. He was a Visiting Scholar with the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 2008, and the University of Pennsylvania, Philadelphia, PA, USA, in 2012, 2014, and 2016. He is the author or a coauthor of more than 70 publications in peer-reviewed conferences and more than 20 publications in impacted journals.