energies

MDPI

Article

On Obtaining Energy-Optimal Trajectories for Landing of UAVs

Dariusz Horla * and Jacek Cieślak

Institute of Robotics and Machine Intelligence, Faculty of Control Robotics and Electrical Engineering, Poznan University of Technology, ul. Piotrowo 3a, 60-965 Poznan, Poland; jacek.cieslak95@gmail.com

* Correspondence: dariusz.horla@put.poznan.pl

Received: 21 March 2020; Accepted: 14 April 2020; Published: 20 April 2020

Abstract: The optimization issues connected to a landing task of an unmanned aerial vehicle are discussed in the paper, based on a model of a mini-class drone. Three landing scenarios are considered, including minimum-time landing, landing with minimum energy consumption, and planned landing. With the use of classical dynamic programming techniques, including the minimum principle of Pontryagin, as well as the calculus of variations, the optimal altitude reference trajectories are found, to form the altitude control system in such a way as to mimic the profile of the reference trajectory by the actual altitude of the UAV. The simulation results conducted with the use of the Simulink Support Package for Parrot Minidrones verify the correctness and effectiveness of the method, and open the research directions for further analysis, especially to tune altitude controller in a way, as to track the reference profile. Up to this point, optimization tasks considered in the literature, with respect to the drones, were connected to swarm formation optimization, optimization of the take-off process or landing process limited to optimal path planning. This paper thus considers a new topic in the field.

Keywords: energy; landing; optimization; unmanned aerial vehicle

1. Introduction

Recent years have brought an increased popularity of unmanned aerial vehicles (UAVs), which, due to their increasing popularity in the market, have wider and wider applicability areas, ranging from pure recreation to scientific research. The latter implies the need to improve control algorithms which govern behaviour of UAVs to increase their safety and reliability. This can also be understood in the context of energy consumption efficiency.

A variety of tasks carried out by UAVs is a challenge for control engineers [1] responsible for controller tuning of drones. During in flight conditions, the UAVs are required to perform both agile, as well as precise maneuvers, also including maneuvers in formations of UAV, making the issues more complicated [2]. However, during the landing process, the situation is different, and the drones should closely abide some reference trajectories to achieve efficient landing with respect to some performance indices. In addition, at any stage of the flight, it might be necessary to land due to environmental issues, or low battery state, and to execute the flying scenario on the basis of, for example, some state machine.

In 2016, the Department and Aeronautics and Astronautics (AeroAstro) from Massachusetts Institute of Technology (MIT) designed the *Simulink Support Package for Parrot Minidrones* as Matlab's add-on [3]. The software enables one to design, simulate and test control strategies using real UAVs, and is dedicated to Rolling Spider and Mambo drones of Parrot. It can be also used, as in the current paper, to perform model-in-the-loop simulations, to verify the core idea of the paper, that is, generation of the optimal reference landing trajectory to achieve optimal energy consumption feature.

This paper aims at finding optimal landing trajectories for different landing scenarios, as well as focusing on giving directions for modifications of the altitude control system, to tune their controllers in such a way as to achieve optimal performance, which will be a next research stage.

In the literature, one can find multiple approaches to landing trajectory optimization; however, they are usually connected either to landing on a moving target, or approaching the landing spot with a non-zero horizontal velocity. This is not the case here, as this approach can be easily thought of as a pick-and-place stage of the landing/take-off procedure.

Various papers give interesting applications to the drones' deployment, using optimization algorithms. For example, in Reference [4] a Hungarian algorithms is used on the basis of bipartie graphs to match drones to nesting stations, with the increasing interest for smart cities applications. In Reference [5], the authors have used reinforcement learning to obtain the solution, where the agent is able to learn the network topology and infer some information about the environment, in order to find the optimal trajectory that lets the UAV autonomously return to its landing spot within the flying time limit. Similarly, the authors of Reference [6] address trajectory planning issues for an aerial platform to identify and land on a moving car. In order to govern the behaviour of the UAV in real time, the model predictive control is adopted for a generated reference trajectory for the UAV, which are then tracked by the nonlinear feedback controller. The problem of flight path planning of the UAV landing on a moving vessel is discussed in Reference [7], where a method for calculating the optimal approach landing trajectory between an UAV and a small vessel is obtained on the basis of an iterative method. In Reference [8], an optimization task of landing swarms of UAVs is given. The paper is cited not in order to compare the approach presented therein, but to show the optimization has been used to find optimal position of the swarm, when reaching target position, not to generate a vertical take-off and landing-like (VTOL) trajectory of landing.

The author of Reference [9] considers the perched landing, on the contrary to a standard landing on a runaway, of a fixed-wing UAV, where the goal is to deliver the vehicle at point just above the runway surface with near-zero vertical velocity and finite horizontal velocity. During a perched landing, at a certain point in space it is required to have nominally zero vertical velocity and zero forward velocity. The authors of Reference [10] propose the optimization approach to obtain reference landing trajectories in an emergency situation for fixed-wing UAVs, and gives a good reference to the 2009 crash at Hudson River.

The authors of Reference [11] consider take-off trajectory optimization problems for a vertical takeoff and landing (VTOL) unmanned aerial vehicle. As in the case of this paper, the longitudinal kinematics equation and dynamics equation are established with atmospheric density, earth gravity, engine thrust and aerodynamic parameters taken into consideration. The objective function of the optimization problem is the fuel consumption and distance at the given time. Unfortunately, this study is performed for take-off procedure only. An increasing interest in VTOL UAVs, leading to the construction of hybrid structures, see Reference [12], might potentially lead to a further research in the topic at optimization in both take-off and landing stages.

As can be seen, a literature review, reveals that there is a need for the solution to the optimal landing task. The paper is structured as follows—Section 2 gives a simplified mathematical model of a quadrotor fitted to the vertical landing task, Section 3 provides details about model reduction to implement this task efficiently, in Section 4 some basic landing issues are presented, and Sections 5–7 describe minimum-time, minimum-energy, and velocity-penalised landing scenarios, respectively. In Section 8, a comparison of the obtained optimal reference trajectories is given. Section 9 gives an insight into the Simulink Support Package for Parrot Minidrones, to enable model-in-the-loop simulation, presenting the behaviour of the closed-loop system to track given reference trajectories, related to optimal performance. Finally, Section 10 gives a short summary of the work.

2. Simplified Mathematical Model of a Quadrotor

Let us describe a model of a Parrot Rolling Spider quadrotor driven by DC motors. In order to define position and orientation of this drone, global and local coordinate systems need to be introduced.

The global system is described in NED convention (North-East-Down), and the local system has its origin in the geometrical center of the drone. Its axes, that is, x and y point towards the two adjacent motors, whereas the axis z points downwards. The orientation of the drone can be described using the Euler angles [13] as listed below. The coordinate system of a UAV and the Euler angles are depicted in Figure 1, with:

- ϕ – angle w.r.t. the x axis (*roll* angle),
- θ – angle w.r.t. the y axis (*pitch* angle),
- ψ – angle w.r.t. the z axis (*yaw* angle).

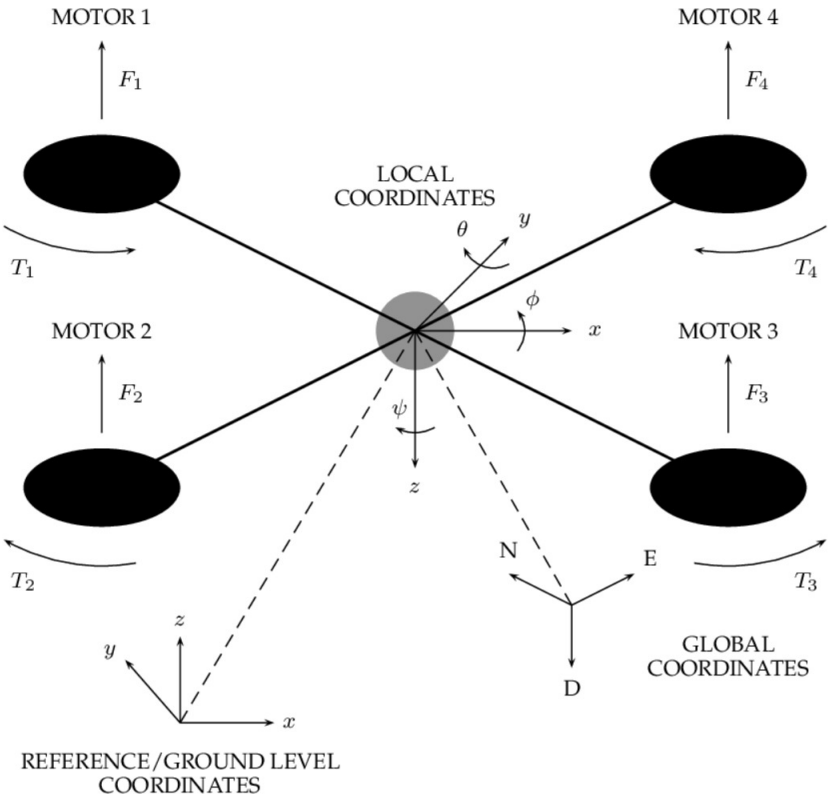


Figure 1. Coordinate systems and Euler angles.

The transformation between global and local coordinates requires one to multiply three rotation matrices $\boldsymbol{R}_{\text{yaw}}$, $\boldsymbol{R}_{\text{pitch}}$ and $\boldsymbol{R}_{\text{roll}}$, respectively, to obtain the final transformation matrix $\boldsymbol{R}_{\text{W2B}}$. In the transformations given below, the following notation has been adopted $s(\alpha) := \sin(\alpha)$, $c(\alpha) := \cos(\alpha)$, and:

$$\begin{aligned}\boldsymbol{R}_{\text{W2B}} &= \boldsymbol{R}_{\text{yaw}}\boldsymbol{R}_{\text{pitch}}\boldsymbol{R}_{\text{roll}} = \\ &= \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c(\theta) & 0 & s(\theta) \\ 0 & 1 & 0 \\ -s(\theta) & 0 & c(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -s(\phi) \\ 0 & s(\phi) & c(\phi) \end{bmatrix} = \\ &= \begin{bmatrix} c(\theta)c(\psi) & c(\psi)s(\theta)s(\phi) - c(\theta)s(\psi) & s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta) \\ c(\theta)s(\psi) & c(\phi)c(\psi) + s(\theta)s(\phi)s(\psi) & c(\phi)s(\theta)s(\psi) - c(\psi)s(\phi) \\ -s(\theta) & c(\theta)s(\phi) & c(\theta)c(\phi) \end{bmatrix}. \end{aligned} \tag{1}$$

The backward transformation from local to global coordinates requires the matrix $\boldsymbol{R}_{\mathrm{W2B}}$ to be inverted,

$$\begin{aligned}\boldsymbol{R}_{\mathrm{B2W}} &= \boldsymbol{R}_{\mathrm{W2B}}^{-1} = \\ &= \begin{bmatrix} c(\theta)c(\psi) & c(\theta)s(\phi) & -s(\theta) \\ c(\psi)s(\theta)s(\phi) - c(\phi)s(\psi) & c(\phi)c(\psi) + s(\theta)s(\phi)s(\psi) & c(\theta)s(\phi) \\ s(\psi)s(\phi) + c(\psi)c(\phi)s(\theta) & c(\phi)s(\psi)s(\theta) - c(\psi)s(\phi) & c(\theta)c(\phi) \end{bmatrix}.\end{aligned} \tag{2}$$

Similarly, the transformation from local rotational speeds $\underline{\Omega} = [p, q, r]^T$ to Euler angle derivatives $\underline{\dot{\Theta}} = [\dot{\phi}, \dot{\theta}, \dot{\psi}]^T$ is possible when the inverse of Wronskian matrix $\boldsymbol{W}$ is used [13]:

$$\underline{\dot{\Theta}} = \boldsymbol{W}^{-1}\underline{\Omega}, \tag{3}$$

$$\boldsymbol{W}^{-1} = \begin{bmatrix} c(\theta) & s(\phi)s(\theta) & c(\phi)s(\theta) \\ 0 & c(\phi)c(\theta) & -s(\phi)c(\theta) \\ 0 & s(\phi) & c(\phi) \end{bmatrix}. \tag{4}$$

According to the second Newton's principle, the acceleration exerted by the unbalanced force $\vec{F_w}$ on a body of a mass m is proportional to this force, and inversely proportional to its mass, that is,

$$\vec{F_w} = \frac{d}{dt}(m\vec{v}). \tag{5}$$

Having assumed that $\vec{F_w} = \vec{G} - \vec{T}$, where $\vec{G}$ is a gravity force vector, and $\vec{T}$ denotes the thrust vector generated by the propellers, Equation (5) can be extended to the form

$$\underline{\dot{v}} = \frac{1}{m}\left(\begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \boldsymbol{R}_{\mathrm{B2W}}\begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}\right). \tag{6}$$

For the sake of brevity, the arrows above symbols will be henceforth omitted, as only numerical values will be considered. By substituting the matrix $\boldsymbol{R}_{\mathrm{B2W}}$ from Equation (2) to Equation (6), it is possible to obtain the acceleration vector in three axes of the local coordinate system of the UAV

$$\underline{\dot{v}} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} -T(c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi)) \\ -T(c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi)) \\ mg - T(c(\phi)c(\theta)) \end{bmatrix}. \tag{7}$$

The thrust T generated by the propellers can be calculated on the basis of formulae below:

$$T_i = K_a\omega_i^2, \tag{8}$$

$$K_a = C_t\rho(h)Ar^2, \tag{9}$$

$$T = K_a\sum_{i=1}^{n}(\omega_i^2), \tag{10}$$

where:

- T_i (N) – thrust of the i–th propeller,
- T (N) – accumulated thrust of all propellers,
- ω_i ($\frac{\mathrm{rad}}{\mathrm{s}}$) – rotational speed of the i–th propeller,
- K_a (kg · m) – aerodynamical constant,
- C_t – thrust constant,
- ρ ($\frac{\mathrm{kg}}{\mathrm{m}^3}$) – air density,

- $h\,(\mathrm{m})$ – propeller height,
- $A\,(\mathrm{m}^2)$ – propeller area,
- $r\,(\mathrm{m})$ – propeller radius,
- n – number of propellers.

The considered quadrotor ($n = 4$) has six degrees of freedom, thus, in order to describe its state, 12 state variables are necessary [13],

$$\begin{aligned} \underline{x} &= [\mathrm{x}_1,\ \mathrm{x}_2,\ \mathrm{x}_3,\ \mathrm{x}_4,\ \mathrm{x}_5,\ \mathrm{x}_6,\ \mathrm{x}_7,\ \mathrm{x}_8,\ \mathrm{x}_9,\ \mathrm{x}_{10},\ \mathrm{x}_{11},\ \mathrm{x}_{12}]^T = \\ &= [x,\ y,\ z,\ \phi,\ \theta,\ \psi,\ \dot{x},\ \dot{y},\ \dot{z},\ p,\ q,\ r]^T. \end{aligned} \tag{11}$$

In Equation (11), the symbol $\underline{x}$ refers to a full state vector, whereas x and $\dot{x}$ are position and velocity of the UAV, respectively, in axis x, and so forth.

3. Model Reduction

The analysis of the landing process requires the model to be simplified. It is assumed that the UAV does not move in XY plane and keeps a horizontal alignment, thus without any loss of generality it can be assumed that:

$$x = \dot{x} = \ddot{x} = y = \dot{y} = \ddot{y} = 0, \tag{12}$$

$$\phi = \theta = 0. \tag{13}$$

This assumption is realistic, and present in multiple pick-and-place tasks, as eventually, the control system of the UAV should keep this alignment. When environmental disturbances take place, the UAV is likely to be moved away from the landing position, but when emergency or planned landing is executed, the precision of landing in a spot is not of prime importance. It is a trajectory of descend the most important factor, thus the precise location of landing is not. Abiding Equations (12) and (13) strictly, would stipulate no exogenous disturbances occur. This simplification is somewhat naturel for tilt-hex constructions, see Reference [14] where the accumulative thrust force can be exerted in non-vertical direction, without tilting the structure, to compensate for the disturbances.

It has also been assumed that all four motors have the same rotational speed, leading to simplifying Equations (7) and (10) to the form of:

$$T = 4K_a\omega^2, \tag{14}$$

$$\ddot{z} = g - \frac{T}{m} = g - 4\frac{K_a}{m}\omega^2. \tag{15}$$

In order to enable any further analysis, the third coordinate system has been introduced, with the origin at the ground level, and the z axis pointing upwards. As the landing takes place while the altitude is not extremally high, it has been assumed that the air density does not vary, then:

$$\rho = \mathrm{const} \Rightarrow K_a = \mathrm{const}. \tag{16}$$

The symbolic expressions α and u have been defined as (:=):

$$\alpha := 4 \cdot \frac{K_a}{m}, \tag{17}$$

$$u := \omega^2, \tag{18}$$

$$T = \alpha u, \tag{19}$$

$$\ddot{z} = \alpha u - g, \tag{20}$$

leading to the introduction of a simplified model of dynamics in the z axis

$$\dot{x}_1 = \ddot{z} = \alpha u - g \tag{21}$$

$$\dot{x}_2 = \dot{z} = x_1 \tag{22}$$

in the third coordinate system, with the output equation (state-space equations) $y = x_2$.

The dynamical model of the UAV can be put in a matrix form (with the time indices omitted for brevity)

$$\underline{\dot{x}} = \boldsymbol{A}\underline{x} + \boldsymbol{B}\underline{u}, \tag{23}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \alpha & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ g \end{bmatrix} \tag{24}$$

with the output equation

$$y = \underline{c}^T \underline{x} = [0, 1]\underline{x} = x_2 \,.$$

As can be seen, it is a typical case of a system which is structurally unstable (as $g \neq 0$ at all times), resembling many other dynamical structures, as a bicycle robot, subject to the same force, see Reference [15].

4. Landing Issues

The landing procedure is different from any control task of the UAV during the main phase of flight. The major difference is that reducing the current altitude to the ground level accepts no overshoots, which are typical in any transient processes during regulation tasks. The other issue is that different landing scenarios should be taken into consideration, as the UAV might undergo emergency landing or might land in a planned way. The first one can be given rise due to, for example, low battery levels or due to environmental issues. All the cases imply the landing trajectory $h_{\text{start}} \rightarrow h_{\text{ground}}$, to vary, where h_{start} denotes the initial altitude of the UAV at the time instant t_0, and h_{ground} denotes the ground level.

The following have been henceforth assumed:

$$z(0) = h_{\text{start}}, \tag{25}$$

$$z(t_f) = \dot{z}(0) = \dot{z}(t_f) = 0, \tag{26}$$

$$z(t) \geq 0, \tag{27}$$

where $z(t)$ denotes the position of the UAV in z axis in the third coordinate system.

Among the others, it is important to classify landing procedures in the following pattern:

- minimum-time landing,
- minimum-energy landing,
- velocity-penalized landing,

as they arise from typical and emergency scenarios which can be potentially considered.

It is to be stressed that the generated reference trajectories should, in principle, be further fed to the altitude control loop to preserve desired behaviour of the UAV.

5. Minimum-Time Landing

The first scenario considered is the landing in the minimum time, which requires the optimization techniques to be adopted, and is connected to the following cost function:

$$J = \int_0^{t_f} (1)\, dt = t_f, \tag{28}$$

where t_f is the duration of the landing procedure (final time), with the assumption that the thrust can be generated upwards only, and the propellers have a limited rotational speed

$$0 \leq u \leq u_{\max}. \tag{29}$$

Following a standard routine for minimum-time problems, the following need to be calculated: $\boldsymbol{G}_i = [\underline{b}_i, \boldsymbol{A}\underline{b}_i, \boldsymbol{A}^2\underline{b}_i, \ldots, \boldsymbol{A}^{n-1}\underline{b}_i]$:

$$\boldsymbol{G}_1 = \begin{bmatrix} \begin{bmatrix} \alpha \\ 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix}, \tag{30}$$

$$\boldsymbol{G}_2 = \begin{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \tag{31}$$

$$\det(\boldsymbol{G}_1) = \alpha^2, \tag{32}$$

$$\det(\boldsymbol{G}_2) = 1. \tag{33}$$

to verify that the determinants of $\boldsymbol{G}_1$, $\boldsymbol{G}_2$ are non zero for $\alpha \neq 0$, and whether the minimum-time control is unique [16]. As known, such a scenario leads to a two-phase control: free falling phase, and braking phase with maximal $u = u_{\max}$ [16]. This is a common solution to minimum-time problems arising in control engineering practice, using for example, the minimum principle. The minimum-time control combined of two phases, makes use of the properties of the system, to allow maximum acceleration increase with no energy lost on rotating the propellers (motors stop), and the braking stage. In this sense, it resembles other approaches identified in the field, to use internal system properties to carry on control tasks, see Reference [17]. Below, t_s denotes the switching time, t_f is the grounding time, and $T_{\max} = \alpha u_{\max}$.

First landing phase: Second landing phase:

$$\begin{cases} u &= 0 \\ \dot{x}_1 &= -g \end{cases} \qquad \begin{cases} u &= T_{\max} \\ \dot{x}_1 &= T_{\max} - g \end{cases}$$

$$x_1(0) = x_1(t_f) = 0 \Rightarrow \int_0^{t_s} (-g)dt + \int_{t_s}^{t_f} (T_{\max} - g)dt = 0,$$

$$0 = -gt_s + (T_{\max} - g)(t_f - t_s),$$

$$gt_s = (T_{\max} - g)t_f - (T_{\max} - g)t_s,$$

$$T_{\max}t_s = (T_{\max} - g)t_f,$$

$$t_f = \frac{T_{\max}}{(T_{\max} - g)}t_s,$$

$$\gamma = \frac{T_{\max}}{(T_{\max} - g)},$$

$$t_f = \gamma t_s.$$

Having assumed that assumptions Equations (25) and (26) hold and $\dot{z}(t_s) = -gt_s$, the velocity at the first phase of the landing procedure satisfies:

$$\begin{aligned}
0 &= h_{\text{start}} + \int_0^{t_s} (-gt)dt + \int_0^{t_s} (\dot{z}(0))dt + \int_{t_s}^{t_f} ((T_{\max} - g)t)dt + \int_{t_s}^{t_f} (\dot{z}(t_s))dt, \\
0 &= h_{\text{start}} - \frac{1}{2}g{t_s}^2 + \frac{1}{2}(T_{\max} - g)(t_f - t_s)^2 + \dot{z}(t_s)(t_f - t_s), \\
-h_{\text{start}} &= -\frac{1}{2}g{t_s}^2 - \frac{1}{2}(g - T_{\max})(t_f - t_s)^2 - gt_s(t_f - t_s), \\
h_{\text{start}} &= \frac{1}{2}g{t_s}^2 + \frac{1}{2}(g - T_{\max})(t_f - t_s)^2 + gt_s(t_f - t_s), \\
h_{\text{start}} &= \frac{1}{2}\left[(g - T_{\max})(\gamma - 1)^2 + g(2\gamma - 1)\right]{t_s}^2, \\
h_{\text{start}} &= \frac{1}{2}\left[(g - T_{\max})\gamma^2 + 2T_{\max}\gamma - T_{\max}\right]{t_s}^2,
\end{aligned}$$

and for $\zeta := \frac{1}{2}\left[(g - T_{\max})\gamma^2 + 2T_{\max}\gamma - T_{\max}\right]$:

$$\begin{aligned}
h_{\text{start}} &= \zeta {t_s}^2, \\
\zeta &= \frac{1}{2}\left[(g - T_{\max})\gamma^2 + 2T_{\max}\gamma - T_{\max}\right], \\
\zeta &= \frac{1}{2}\left[-(T_{\max} - g)\frac{T^2}{(T_{\max} - g)^2} + \frac{2{T_{\max}}^2}{(T_{\max} - g)} - \frac{T_{\max}(T_{\max} - g)}{(T_{\max} - g)}\right], \\
\zeta &= \frac{1}{2}\left[\frac{-{T_{\max}}^2 + 2T_{\max}^2 - {T_{\max}}^2 + gT_{max}}{(T_{\max} - g)}\right], \\
\zeta &= \frac{gT_{\max}}{2(T_{\max} - g)}, \\
h_{\text{start}} &= \zeta {t_s}^2 \Rightarrow t_s = \sqrt{\frac{h_{\text{start}}}{\zeta}}.
\end{aligned}$$

The switching t_s and grounding t_f times satisfy:

$$t_s = \sqrt{\frac{2h_{\text{start}}(T_{\max} - g)}{gT_{\max}}}, \tag{34}$$

$$t_f = \frac{T_{\max}}{(T_{\max} - g)}t_s, \tag{35}$$

$$t_f = \sqrt{\frac{2h_{\text{start}}T_{\max}}{g(T_{\max} - g)}}. \tag{36}$$

On the basis of Equations (34) and (36), it is possible to provide the analytical formula for the landing trajectory

$$z(t) = x_2^*(t) = \begin{cases} h_{\text{start}} - \frac{1}{2}gt^2 & \Leftrightarrow \quad 0 \le t < t_s \\ h_{\text{start}} - \frac{1}{2}g{t_s}^2 - gt_s(t - t_s) + \frac{1}{2}(T_{\max} - g)(t - t_s)^2 & \Leftrightarrow \quad t_s \le t < t_f. \\ 0 & \Leftrightarrow \quad t \ge t_f \end{cases} \tag{37}$$

Sample landing trajectories for the Craig model have been presented in Figure 2 with the parameters of the Parrot Rolling Spider UAV and various initial altitudes:

$$\begin{aligned} C_t &= 0.0107, \\ \rho &= 1.184\,\frac{\mathrm{kg}}{\mathrm{m}^3}, \\ r &= 0.033\,\mathrm{m}, \\ m &= 0.068\,\mathrm{kg}, \\ \omega_{\max} &= 2630.67\,\frac{1}{\mathrm{s}}. \end{aligned}$$

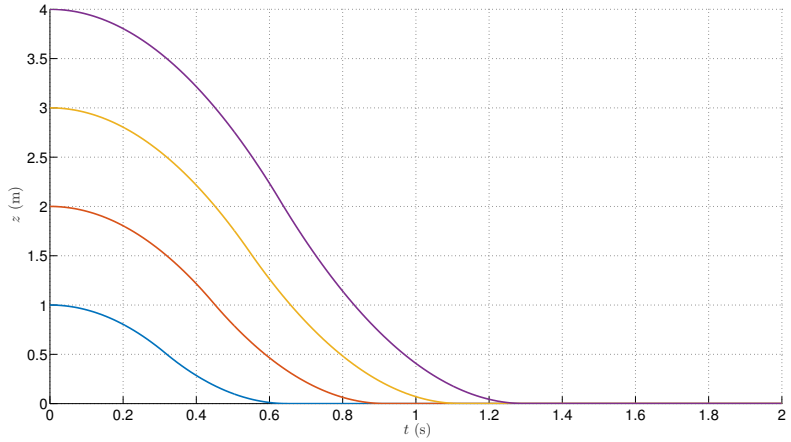


Figure 2. Minimum-time trajectories (for various initial altitudes—see the values at $t = 0$).

6. Minimum-Energy Landing (False-zero Landing)

Minimum-energy landing is connected to the minimization of the performance index taking both altitude and control u effort (energy) into account, with the weight β and

$$J = \int_0^t \left(x_2 + \beta u^2 \right) dt, \tag{38}$$

leading to the following optimization task:

$$\min_u J \tag{39}$$

$$\text{s.t.} \quad \dot{x}_1 = \alpha u - g \tag{40}$$

$$\dot{x}_2 = x_1 \tag{41}$$

The task has been solved using the minimum principle of Pontryagin by minimizing the following Hamiltonian function [16,18]:

$$H = {x_2}^2 + \beta u^2 + p_1(\alpha u - g) + p_2 x_1. \tag{42}$$

By transforming Equation (21) one gets:

$$\begin{aligned}
\dot{x} &= \alpha u - g,\\
u &= \frac{1}{\alpha}(\dot{x_1} + g),\\
\dot{u} &= \frac{1}{\alpha}\ddot{x_1},\\
\ddot{u} &= \frac{1}{\alpha}{x_1}^{(3)}.
\end{aligned}$$

Using the terminal condition $H^*(t) = H^*(t_f) = 0$, and subject to no constraints imposed on $u(t)$:

$$\begin{aligned}
\frac{\delta H}{\delta u} &= 2\beta u + p_1\alpha = 0,\\
p_1 &= -\frac{2\beta}{\alpha}u,\\
\dot{p_1} &= -\frac{2\beta}{\alpha}\dot{u},\\
\ddot{p_1} &= -\frac{2\beta}{\alpha}\ddot{u},\\
\dddot{p_1} &= -\frac{2\beta}{\alpha}\frac{1}{\alpha}{x_1}^{(3)},\\
\dddot{p_1} &= -\frac{2\beta}{\alpha^2}{x_1}^{(3)},
\end{aligned}$$

with:

$$\begin{aligned}
\dot{p_1} &= -\frac{\delta H}{\delta x_1} = -p_2,\\
\dot{p_2} &= -\frac{\delta H}{\delta x_2} = -2x_2,\\
\ddot{p_1} &= 2x_2,\\
0 &= \frac{\beta}{\alpha^2}{x_1}^{(3)} + x_2,\\
C &:= \frac{\beta}{\alpha^2},\\
0 &= C{x_1}^{(3)} + x_2|\frac{d}{dt},\\
0 &= C{x_1}^{(4)} + x_1,
\end{aligned}$$

$$\begin{aligned}
x_1 &= e^{rt},\\
0 &= Ce^{rt(4)} + e^{rt},\\
0 &= Cr^4e^{rt} + e^{rt}| : e^{rt},\\
0 &= Cr^4 + 1,\\
\psi &:= \frac{\sqrt{2}}{2\sqrt[4]{C}},\\
r &= \begin{cases}
\psi(1+j)\\
\psi(1-j)\\
-\psi(1+j)\\
-\psi(1-j).
\end{cases}
\end{aligned}$$

A general formula describing $x_1(t)$, $x_2(t)$ is given as:

$$\begin{aligned}
x_1(t) =& e^{\psi t}(C_1 c(\psi t) + C_2 s(\psi t)) + e^{-\psi t}(C_3 c(\psi t) + C_4 s(\psi t)),\\
x_2(t) =& \int (x_1(t))dt,\\
x_2(t) =& C_1\left(\frac{e^{\psi t}}{2\psi^2}(\psi c(\psi t) + \psi s(\psi t))\right) + C_2\left(\frac{e^{\psi t}}{2\psi^2}(\psi s(\psi t) - \psi c(\psi t))\right) +\\
&+ C_3\left(\frac{e^{-\psi t}}{2\psi^2}(-\psi c(\psi t) + \psi s(\psi t))\right) + C_4\left(\frac{e^{-\psi t}}{2\psi^2}(-\psi s(\psi t) - \psi c(\psi t))\right),\\
x_2(t) =& \frac{1}{2\psi}\left[e^{\psi t}\left((C_1 - C_2)c(\psi t) + (C_1 + C_2)s(\psi t)\right) +\right.\\
&\left.+ e^{-\psi t}\left((-C_3 - C_4)c(\psi t) + (C_3 - C_4)s(\psi t)\right)\right] + C_5.
\end{aligned} \tag{43}$$

Using assumptions Equations (25) and (26), the integral constants can be found:

$$\begin{aligned}
&\left.\begin{aligned} &x_1(0) = 0 \Rightarrow C_1 + C_3 = 0\\ &x_1(t_f \to \infty) = 0 \Rightarrow C_1 = C_2 = 0\end{aligned}\right\} \quad \Rightarrow C_3 = 0\\
&x_2(t_f \to \infty) = 0 \Rightarrow C_5 = 0,\\
&x_2(0) = h_{\text{start}},\\
&h_{\text{start}} = \frac{1}{2\psi}(-C_3 - C_4) + C_5,\\
&C_4 = -2\psi h_{\text{start}},\\
&x_2(t) = h_{\text{start}} e^{-\psi t}(c(\psi t) + s(\psi t)).
\end{aligned} \tag{44}$$

Finally, the formula describing trajectories depicted in Figure 3 has been obtained. As can be seen, the altitude while landing, falls below zero due to the overshoot phenomenon, what is not acceptable, though definitely shortens the regulation time (typical regulation time vs. damping ratio interplay). In order to compensate for the latter, it has been established that $x_2(t)$ reaches the ground level at $t = \frac{\pi}{\psi}$, and the landing phase can be splitted into two phases:

$$x_2(t) = \begin{cases} f(t) & \Leftrightarrow \quad 0 < t \leq \frac{\pi}{\psi} \\ 0 & \Leftrightarrow \quad t > \frac{\pi}{\psi} \end{cases}, \tag{45}$$

where $f(t)$ results from Equation (43) and reduces the altitude between $h_{\text{start}} \to 0$. In order to eliminate the impact of the overshoot on the landing phase, the ground level should be virtually raised so that the minimum of Equation (43) equals always zero, what explains the 'false zero' name of the approach. The latter has been shown in Figure 4.

The latter implies that the conditions Equations (25) and (26) must be satisfied at all times where:

$$\begin{array}{lll} x_1(0) = 0 & x_1(\frac{\pi}{\psi}) = 0 & x_1(t_f \to \infty) = 0 \\ x_2(0) = h_{\text{start}} & x_2(\frac{\pi}{\psi}) = 0 & x_2(t_f \to \infty) = h_e \end{array}. \tag{46}$$

From relations Equation (46) one gets:

$$\left.\begin{array}{l} x_1(0) = 0 \Rightarrow C_1 + C_3 = 0 \\ x_1(t_f \to \infty) = 0 \Rightarrow C_1 = C_2 = 0 \end{array}\right\} \Rightarrow C_3 = 0,$$

$$x_2(t_f \to \infty) = h_e \Rightarrow C_5 = h_e,$$

$$x_2(0) = h_{\text{start}} \Rightarrow h_{\text{start}} = \frac{-C_4}{2\psi} + h_e,$$

$$C_4 = -2\psi(h_{\text{start}} - h_e),$$

$$x_2(\frac{\pi}{\psi}) = 0 \Rightarrow 0 = -\frac{e^{-\pi}}{2\psi}(-C_4) + h_e,$$

$$e^{-\pi}(h_{\text{start}} - h_e) = h_e,$$

$$h_{\text{start}} = h_e(1 + e^{\pi}),$$

$$h_e = \frac{h_{\text{start}}}{(1 + e^{\pi})},$$

$$C_4 = -2\psi h_{\text{start}}(1 - \frac{h_{\text{start}}}{(1 + e^{\pi})}),$$

$$x_2^*(t) = \begin{cases} h_{start}\left(e^{-\psi t}(1 - \frac{h_{\text{start}}}{(1+e^{\pi})})(c(\psi t) + s(\psi t)) + (1 - \frac{h_{\text{start}}}{(1+e^{\pi})})\right) & \Leftrightarrow \quad 0 < t \leq \frac{\pi}{\psi} \\ 0 & \Leftrightarrow \quad t > \frac{\pi}{\psi} \end{cases}. \tag{47}$$

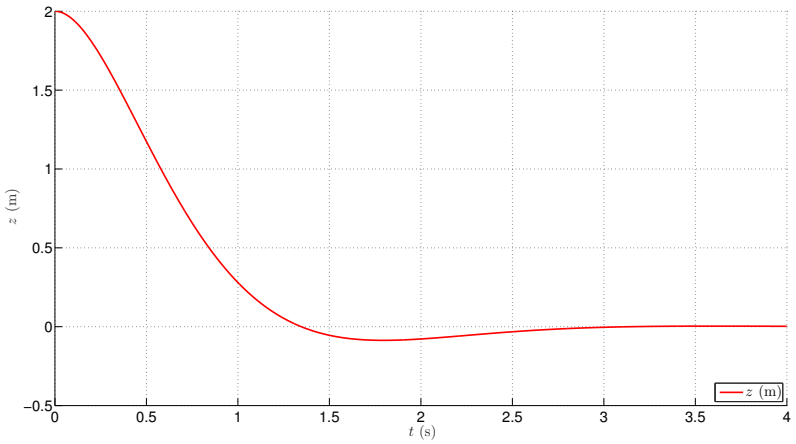


Figure 3. Minimum-energy trajectories—overshoot phenomenon.

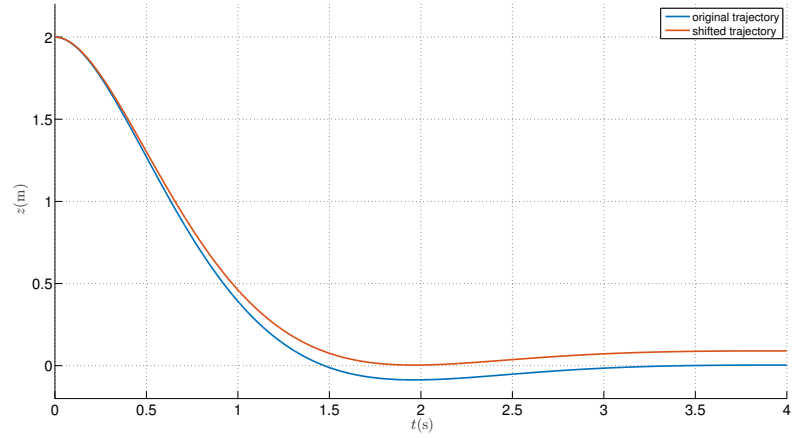
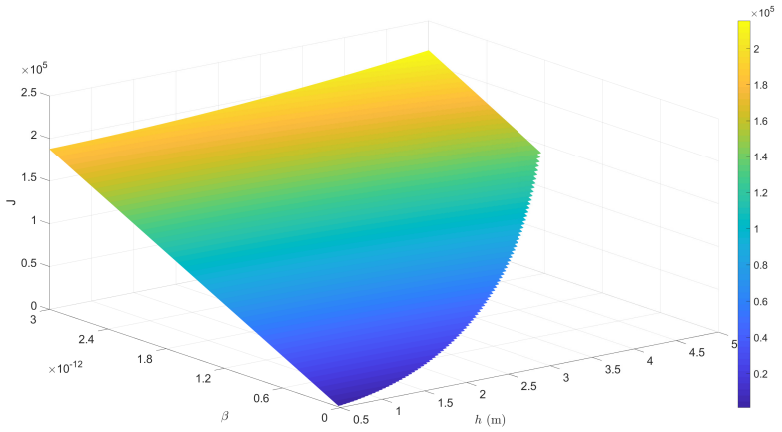
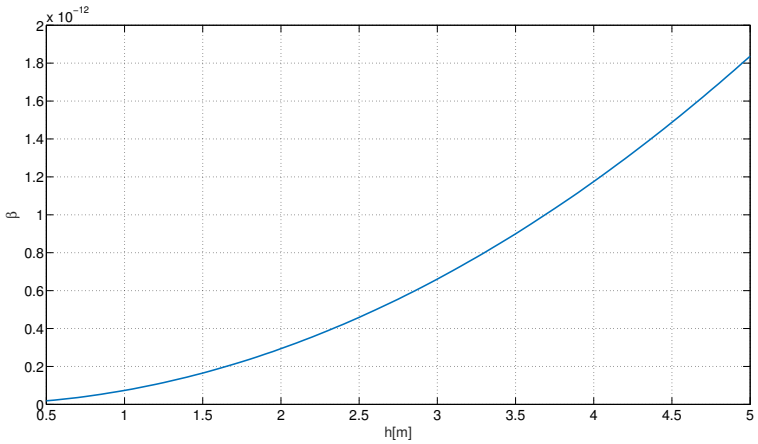


Figure 4. Virtual ground level shift.

It is to be borne in mind that in order to use the trajectory evaluated, the correct value of β must be stipulated first, and subsequently used in the performance index Equation (38). The evaluation of this parameter can be done iteratively by creating, at first, two vectors, which: include a set of initial altitudes, say ranging between 0.5 m up to 5 m with the step of 0.05 m, and the vector of 4000 equally-spaced samples of candidate β values in the range of $\langle 0; 3 \times 10^{-12} \rangle$. The choice of the final value of the parameter is based on evaluation of various trajectories for every pair of the above-mentioned parameters, according to relation Equation (38). Secondly, the samples with negative thrust force need to be removed, and finally, the value of β is based on the minimal value of J for consecutive initial altitude. The results of such iterative calculations is presented in Figure 5.



(**a**) Performance index values



(**b**) Function $\beta(h)$

Figure 5. β values as a function of an altitude h.

The function implementing the calculations in Matlab has been presented in Appendix A (Algorithm A1).

Once the final formula for the landing trajectory $x_2(t)$ is found, and accompanied by the optimal β, it is possible to calculate the trajectories for a specific test conditions. Sample trajectories have been presented in Figure 6.

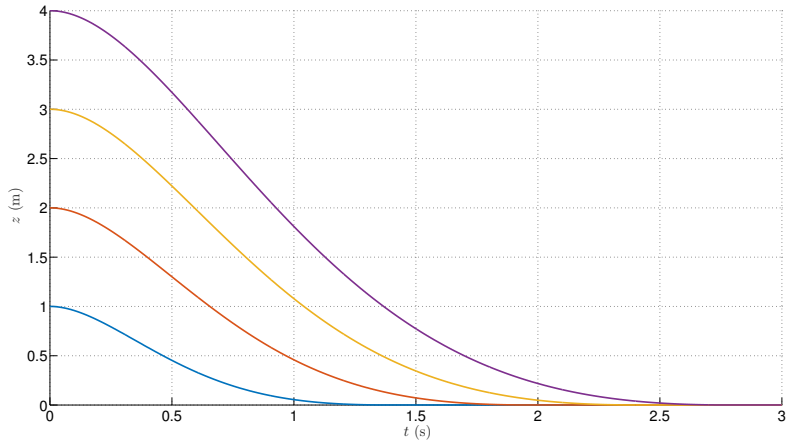


Figure 6. Minimum-energy landing trajectories (for various initial altitudes—see the values at $t = 0$).

7. Velocity-Penalized Landing

The trajectories presented in Sections 5 and 6 can be used during emergency landing, when either time or energy considerations are of prime importance. In these cases, the landing process is fast, and coupled with a huge jerk or snap. However, during the planned landing phase it is useful to take the performance index penalizing the velocity of a drone in the z axis and taking energy considerations into account. During the calculations, a variational calculus approach with constraints is adopted [16], and a penalty taking control signal energy is included into the performance index, to shorten the landing time,

$$J = \int_0^t ({x_1}^2 + \beta u^2) dt. \tag{48}$$

It is necessary to transform the cost function Equation (48) using Equations (21) and (22) into:

$$J = \int_0^t \left({x_1}^2 + \beta u^2\right) dt = \int_0^t \left(\dot{x}_2^2 + \beta \left(\frac{\dot{x}_1}{\alpha} + \frac{g}{\alpha}\right)^2\right) dt =$$
$$= \int_0^t \left(\dot{x}_2^2 + \frac{\beta}{\alpha} (\dot{x}_1 + g)^2\right) dt.$$

Having assumed that $\gamma := \frac{\beta}{\alpha}$,

$$J = \int_0^t \left(\dot{x}_2^2 + \gamma \left(\dot{x}_1^2 + 2g\dot{x}_1 + g^2\right)\right) dt =$$
$$= \int_0^t \left(\dot{x}_2^2 + \gamma \ddot{x}_2^2 + 2g\gamma \ddot{x}_2 + \gamma g^2\right) dt. \tag{49}$$

In order to find the extremum of the functional Equation (49), it is necessary to satisfy the following Euler-Poisson equations for all the functions which are presented in Reference [18]:

$$\frac{\delta f}{\delta x_i} - \frac{d}{dt}\left(\frac{\delta f}{\delta \dot{x}_i}\right) + \frac{d^2}{dt^2}\left(\frac{\delta f}{\delta \ddot{x}_i}\right) - \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\delta f}{\delta {x_i}^{(n)}}\right) = 0, \tag{50}$$

where ${x_i}^{(n)}$ stands for the n-th order derivative of x_i with respect to time. Abiding standard Euler notation, the sub-derivative expressions take the following form: for $n = 1$ one gets $\frac{\delta f}{\delta \dot{x}_i}$, for $n = 2$, $\frac{\delta f}{\delta \ddot{x}_i}$, and so forth.

On the basis of the sub-integral function in Equation (49), a Lagrangean must be defined at first. The term γg^2 has not been included in the performance index, as it is a constant value and has no impact on the precise localisation of the extremum of the functional. In addition, the Lagrange multiplier λ is included into the formula and results from the assumption Equation (27) $x_2 \geq 0 \Rightarrow -x_2 + \kappa^2 = 0$:

$$\begin{aligned} L(t, x_2, \dot{x}_2, \ddot{x}_2, \lambda, \kappa) &= {\dot{x}_2}^2 + \gamma {\ddot{x}_2}^2 + 2g\gamma\ddot{x}_2 + \lambda(-x_2 + \kappa), \\ L &= 0. \end{aligned} \tag{51}$$

In order to formulate the Euler-Poisson equation, it is necessary to calculate the following derivatives of L:

$$\begin{aligned} \frac{\delta L}{x_2} &= -\lambda, & & \\ \frac{\delta L}{\dot{x}_2} &= 2\dot{x}_2, & \frac{d}{dt}\left(\frac{\delta L}{\dot{x}_2}\right) &= 2\ddot{x}_2, \\ \frac{\delta L}{\ddot{x}_2} &= 2\gamma g + 2\gamma\ddot{x}_2, & \frac{d^2}{dt^2}\left(\frac{\delta L}{\ddot{x}_2}\right) &= 2\gamma {x_2}^{(4)}, \\ \frac{\delta L}{\delta\lambda} &= -x_2 + \kappa^2, & \frac{d}{dt}\left(\frac{\delta L}{\lambda}\right) &= 0, \\ \frac{\delta L}{\delta\kappa} &= 2\lambda\kappa, & \frac{d}{dt}\left(\frac{\delta L}{\kappa}\right) &= 0. \end{aligned}$$

On the basis of the latter, and Equation (50), the final set the necessary conditions of optimality is formulated:

$$\begin{cases} 0 &= -\lambda - 2\ddot{x}_2 + 2\gamma {x_2}^{(4)} \\ 0 &= -x_2 + \kappa^2 \\ 0 &= 2\lambda\kappa \end{cases}. \tag{52}$$

Two cases need to be considered here, namely $\lambda \neq 0$ and $\lambda = 0$:

Case I – constraint Equation (27) is active ($z = 0$):

$$\begin{aligned} &\begin{cases} \lambda &\neq 0 \\ \lambda\kappa &= 0, \end{cases} \\ &\kappa = 0, \\ &0 = -x_2 + \kappa^2, \\ &x_2 = 0. \end{aligned} \tag{53}$$

Case II – constraint Equation (27) is inactive ($z > 0$):

$$\begin{cases} \lambda &= 0 \\ \lambda\kappa &= 0 \end{cases} \Rightarrow \begin{cases} \kappa \in \Re \\ x_2 \in \langle 0, \infty \rangle \end{cases},$$

$$0 = -2\ddot{x}_2 + 2\gamma x_2{}^{(4)},$$

$$0 = \gamma \ddot{x}_2{}^{(4)} - \ddot{x}_2,$$

$$x_2 = e^{rt},$$

$$0 = \gamma \left(e^{rt}\right)^{(4)} - \left(e^{rt}\right)^{(2)},$$

$$0 = \gamma r^4 - r^2,$$

$$0 = r^2 \left(\gamma r^2 - 1\right),$$

$$\psi := \sqrt{\frac{1}{\gamma}},$$

$$x_2 = C_1 t + C_2 + e^{-\psi t} \left(C_3 c(\gamma t) + C_4 s(\gamma t)\right). \tag{54}$$

By using Equation (22), a time derivative of Equation (54) can be calculated:

$$x_1 = \frac{d}{dt}(x_2), \tag{55}$$

$$x_1 = C_1 - \psi e^{-\psi t} \left(C_3 c(\gamma t) + C_4 s(\gamma t)\right) + e^{-\psi t} \left(-\gamma C_3 s(\gamma t) + \gamma C_4 c(\gamma t)\right),$$

$$x_1 = C_1 + C_3 e^{-\psi t} \left(-\psi c(\gamma t) - \gamma s(\gamma t)\right) + C_4 e^{-\psi t} \left(-\psi s(\gamma t) + \gamma c(\gamma t)\right). \tag{56}$$

On the basis of the expressions from Equations (25), (26), (54) and (56), the integral constants $C_{1 \div 4}$ can be obtained:

$$\left.\begin{array}{l} x_1(0) = 0 \Rightarrow C_1 + C_3(-\psi) + C_4\gamma = 0 \\ x_1(t_f \to \infty) = 0 \Rightarrow C_1 = 0 \end{array}\right\} \Rightarrow C_4 = \frac{\psi}{\gamma} C_3,$$

$$\left.\begin{array}{l} x_2(0) = h_{\text{start}} \Rightarrow C_2 + C_3 = h_{\text{start}} \\ x_2(t_f \to \infty) = 0 \Rightarrow C_2 = 0 \end{array}\right\} \Rightarrow C_3 = h_{\text{start}},$$

$$C_4 = \frac{\psi}{\gamma} h_{\text{start}},$$

$$x_1^*(t) = h_{\text{start}} \left(e^{-\psi t} \left(-\psi c(\gamma t) - \gamma s(\gamma t)\right) + \frac{\psi}{\gamma} e^{-\psi t} \left(-\psi s(\gamma t) + \gamma c(\gamma t)\right)\right), \tag{57}$$

$$x_2^*(t) = h_{\text{start}} e^{-\psi t} \left(c(\gamma t) + \frac{\psi}{\gamma} s(\gamma t)\right). \tag{58}$$

As in the case of minimum-energy landing, β should also be optimized using, preferably, the same method, though in this case $\beta \in \langle 0; 2 \times 10^{-6} \rangle$, and the cost function takes the form of Equation (48). The listing of a function implementing this idea is given in Appendix A (Algorithm A1). Figure 7 presents velocity-penalized landing trajectories for various initial altitudes.

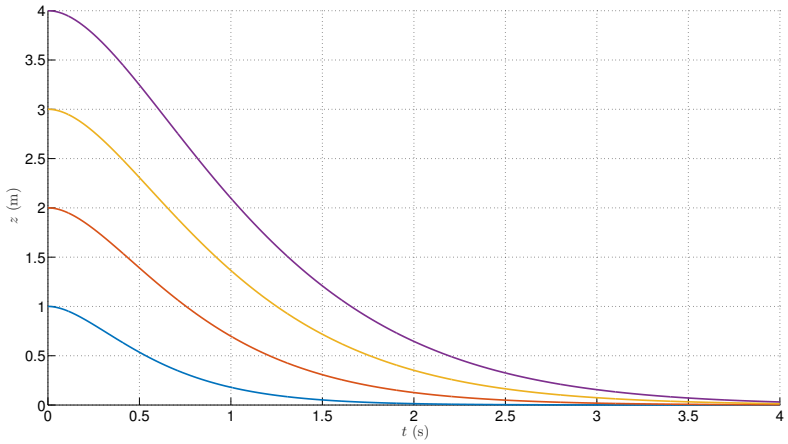


Figure 7. Velocity-penalized trajectories (for various initial altitudes—see the values at $t = 0$).

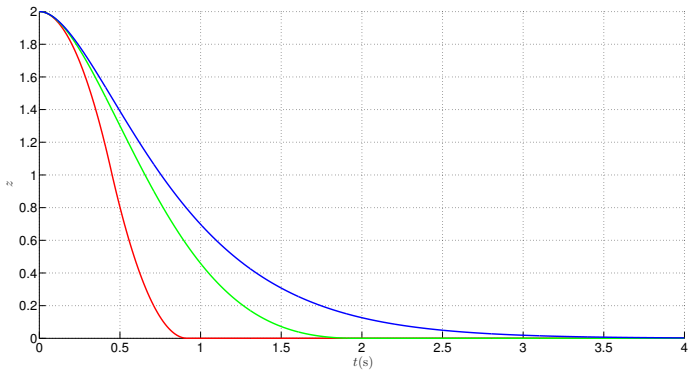
8. Comparison of the Obtained Trajectories

The trajectories obtained in Sections 5–7 have been presented in Figure 8. As can be seen, the time of the landing phase and velocities differ as the function of the selected landing scenario. Having assumed that landing is terminated whenever the difference between the ground level and the current altitude does not exceed 0.01 m, the results have been given in Table 1. The value 0.01 m is arbitrary, and is reflected by the way the real Rolling Spider UAV is affected by disturbances, resulting from the air flow, and poor precision of its inertial measurement unit (IMU).

The methodology behind selecting the gain for the integral term in the PID controller is depicted in Figure 9. By comparing the experiments for various values of the I term gain, the value of 0.01 was selected to mimic relatively fast response time in hover state, while keeping good damping ratio of the control loop. It is a common practice in UAV that controllers actually act like PD controllers, when the I term has low gains.

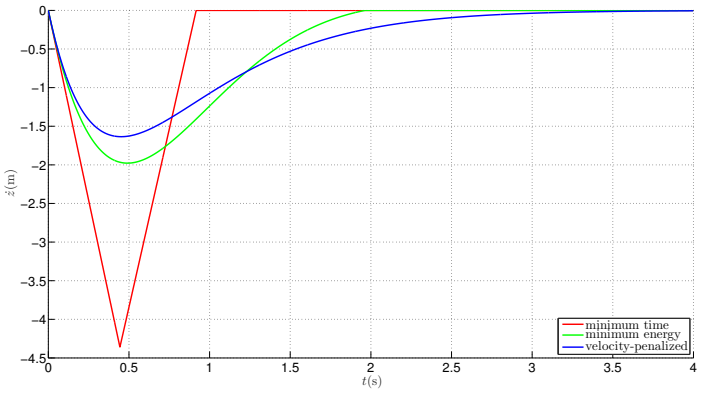
Table 1. Comparison of trajectories.

Trajectory	Landing Time to $z < 0.01$ m(s)	Minimal Altitude (m)
Minimum-time	0.871	4.4409×10^{-17}
Minimum-energy	1.768	5.4839×10^{-9}
Velocity-penalized	3.317	2.5135×10^{-3}

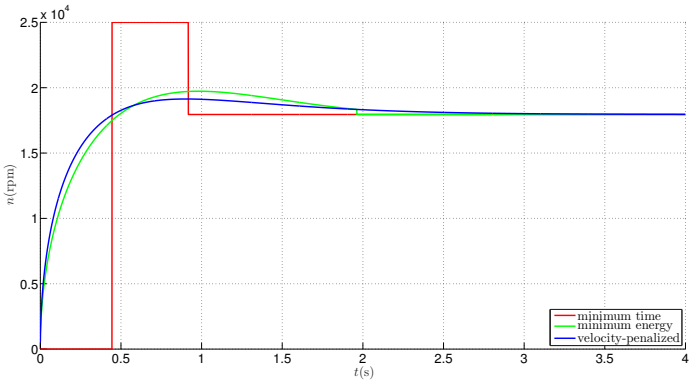


(**a**) Comparison of landing trajectories

Figure 8. *Cont.*



(**b**) Velocities



(**c**) Rotational speed of propellers

Figure 8. Comparison of a landing process.

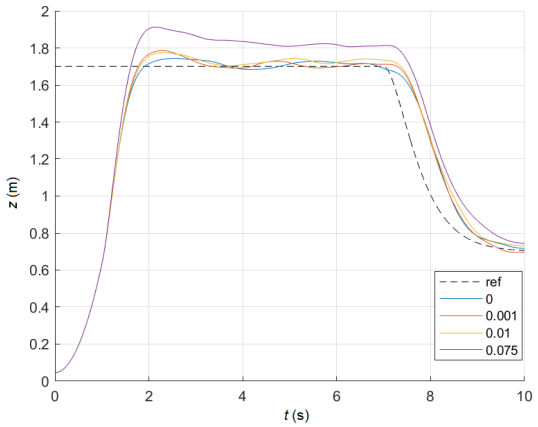
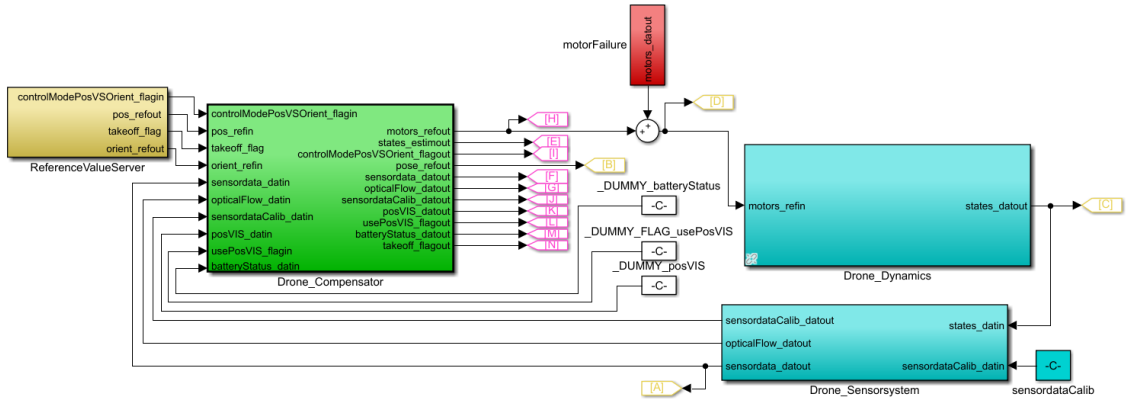


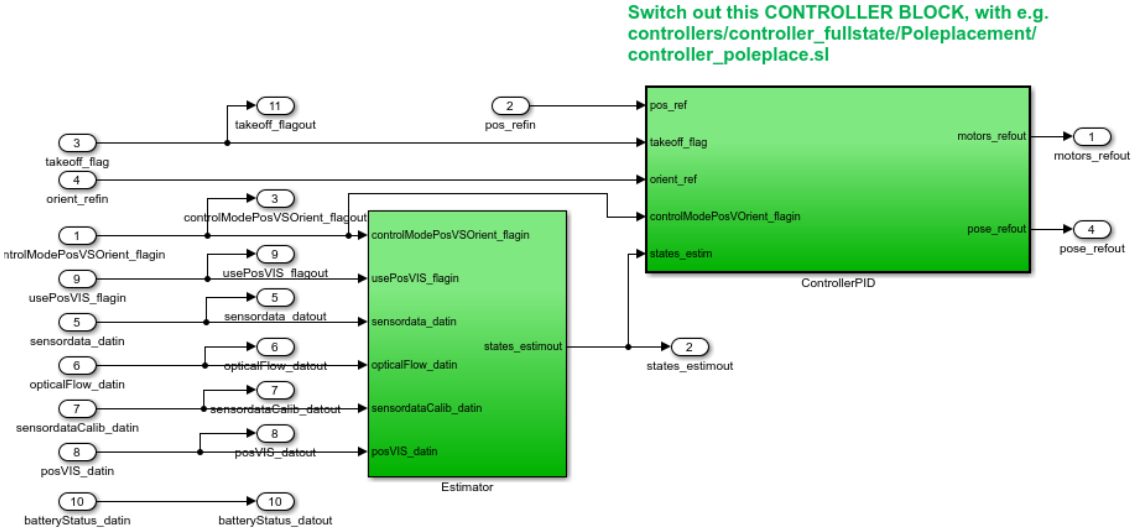
Figure 9. Performance of the altitude stabilization for various integral gains of the PID controller.

9. Simulink Support Package for Parrot Minidrones

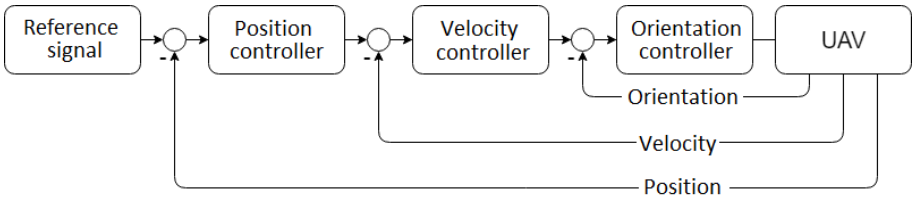
By using the Simulink Support Package for Parrot Minidrones in Matlab is was possible to simulate the behaviour of the drone in safe conditions, and to verify the obtained optimization results. The software enables one to obtain access to all the signals that would be available from the real machine, with the block diagram in the Simulink given in Figure 10.



(**a**) A general block diagram of the control system



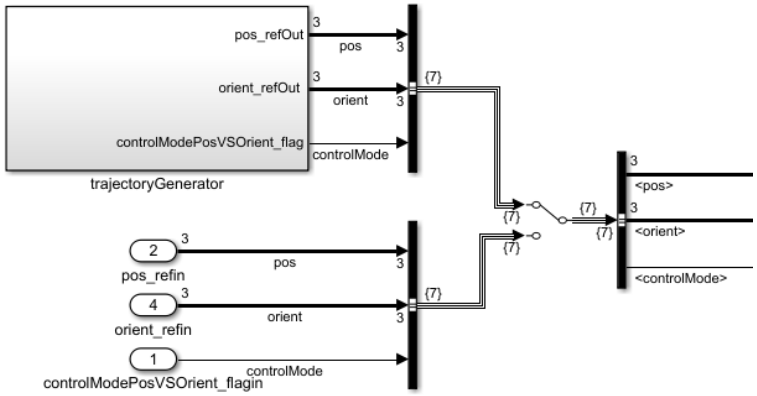
(**b**) Drone_Compensator



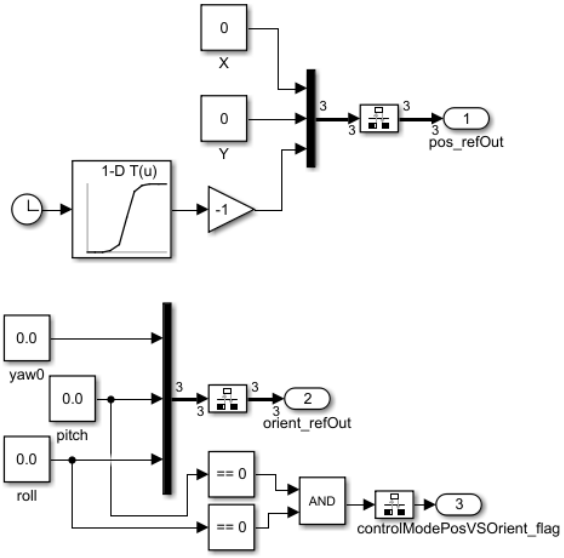
(**c**) Block diagram in the tracking task

Figure 10. Matlab model of the Rolling Spider.

The basic block diagram has been modified so that it is possible to provide the reference trajectory, see Figure 11. Its block diagram depicted using standard control blocks, is presented in Figure 10c. The position signal is connected here to the altitude component which is fed back from the estimator, see Reference [19]. This model, as well as many other mathematical models have been made available to the public by Mathematical Model Database, see Reference [20]. As the Simulink model is implemented in NED convention, the sign of the z coordinate in the Simulink block diagram has to be changed to the opposite.



(**a**) Replacing the original reference with the generator



(**b**) Source of the reference signal

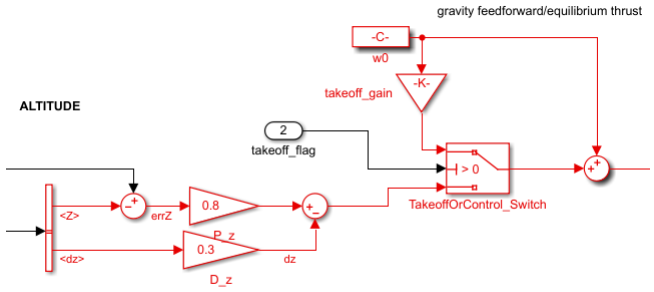
Figure 11. Building a signal generator to provide reference to the drone.

The control system provided by the developer contains the altitude, XY position, yaw, roll and pitch controllers. According to Equation (7) the translation in z axis in independent from the movement in the horizontal plane, thus the modification of the altitude controller is possible without any changes in the remaining controllers.

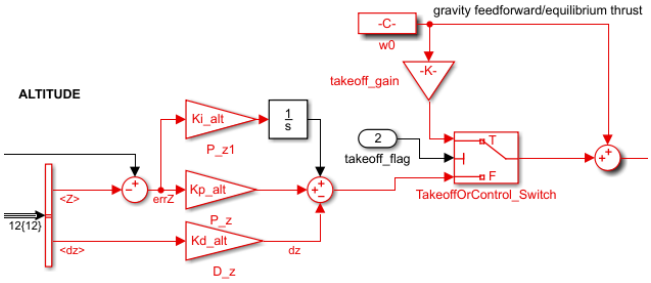
In order to eliminate the steady-state error, an integral action has been added to the original PD controller, with the integral gain set as $K_I = 0.01$, see Figure 12, where the controller is given by ($e(t) = z_{\text{ref}}(t) - z(t)$, as the tracking error, and $z_{\text{ref}}(t)$ as the reference signal),

$$u(t) = k_P e(t) + k_D \frac{de}{dt}, \tag{59}$$

$$u(t) = k_P e(t) + k_I \int e(t)dt + k_D \frac{de}{dt}. \tag{60}$$



(**a**) Primary controller structure



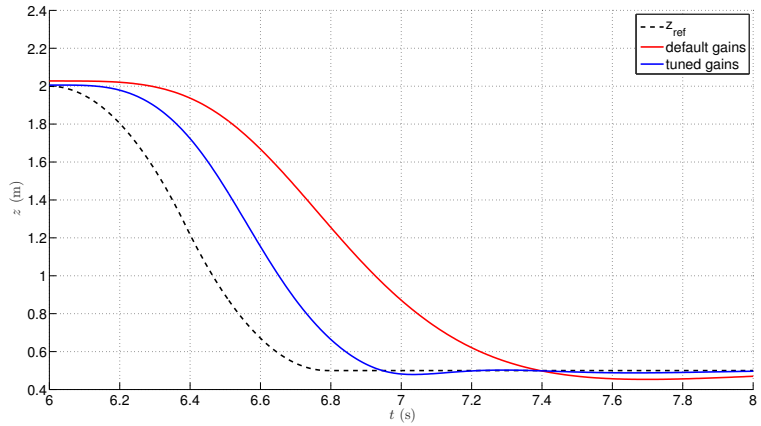
(**b**) Modified controller structure

Figure 12. Primary and modified altitude controller structures.

In this configuration of the controller, P and D gains of the altitude controller have been found by a systematic search within the admissible values' ranges, repeating the experiment to ascend to 2 m, hover, and descend the drone starting at 6 s from the altitude of 2 m to 0.5 m, to observe possible overshoot effects. The gain values have been given in Table 2. The corresponding trajectories from the closed-loop system are presented in Figure 13.

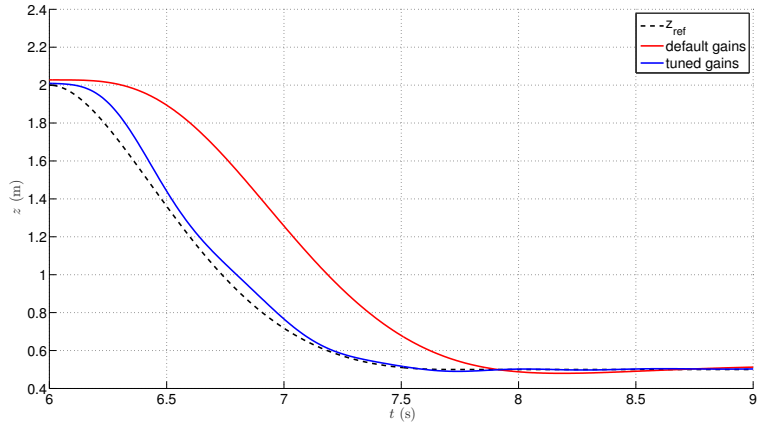
Table 2. The gains used during the simulation.

Gains	Minimum-time	Minimum-energy	Velocity-penalized
K_p	3.695	6.467	4.424
K_d	0.645	0.376	0.259

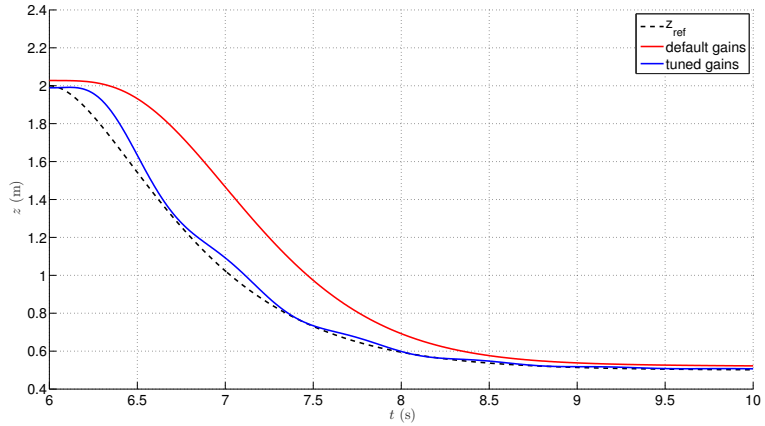


(**a**) Minimum-time trajectory

Figure 13. *Cont.*



(**b**) Minimum-energy trajectory



(**c**) Velocity-penalized trajectory

Figure 13. Comparison of behaviour of the model of the Parrot Rolling Spider in Matlab environment.

10. Summary

In this paper, we demonstrated the applicability of the idea of optimization of the landing process of the UAV to the model-in-the-loop simulation of the Rolling Spider drone. The encouraging performance of the model of the system for various optimization criteria is shown through simulations. The obtained results imply that it is possible to design the control loop in such a way, so that various frequency responses of the closed-loop systems can be obtained, to shape the system's response to a specific altitude reference signal. In order to do the latter, the altitude controller parameters need to be identified to ensure optimal-like behaviour.

In the future work, the authors plan to move from model-in-the-loop to hardware-in-the-loop experiments, using a motion capture system to precisely measure the altitude of the Rolling Spider drone, in comparison to the reference altitude. Such a knowledge enables one conduct further experiments with more complex approaches, including auto-tuning for real UAV experiment. Carrying out such experiments would make the method portable in the sense that it could also work for other types of UAVs, as just a reference profile and the tuning method need to be used.

On the other hand, it is to be stressed that for unstable structures, as presented in Reference [21], it is possible to build a composite feedback linearization control scheme where using an inverse dynamic model a reference trajectory generator is built, and the polynomial approximation of the trajectory is utilized, what calls for multivariable nonlinear programming problems to arise, to fit the polynomial into the waypoints. This might also be an interesting step, requiring good identification of the model,

at first. As a wider and wider spectrum of the UAVs being constructed and successfully deployed, Reference [22], including a spectrum of control approaches, potentially enables researchers to identify new, optimal, control approaches to increase the performance of the flying structures at various stages of flight, including optimal landing procedure.

Author Contributions: Conceptualization, D.H.; methodology, D.H., J.C.; software, J.C.; validation, D.H.; formal analysis, D.H.; investigation, J.C.; writing—original draft preparation, D.H., J.C.; writing—review and editing, D.H.; visualization, J.C.; funding acquisition, D.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Poznan University of Technology grant number 0214/SBAD/0210, and the APC was funded by Poznan University of Technology.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A

Algorithm A1: Optimal β calculation

```
Require: Ts and trajGeneration and trajectory
return BetaSet and J_mat and bbeta and hh
% Ts - sufficiently small sampling period [s]
% trajGeneration - handle to the function generating trajectories
% trajectory - enum variable (trajectory identification)
m=0.068;
Tsim = 5;
switch trajectory
  case cTrajectory.FalseZero
    bbeta = linspace(0,3e-12,4000);
  case cTrajectory.SpeedPenatly
    bbeta = linspace(0,2e-6,4000);
end
hh = 0.5:0.05:5;
J_mat = zeros(length(bbeta), length(hh));
for b_index = 1 to length(bbeta) do
  for h_index = 1 to length(hh) do
    [J_mat(b_index, h_index),TempMat] = ...
    trajGeneration(hh(h_index), m, Tsim, Ts, bbeta(b_index));
    nn = TempMat(:,5);
    if any(nn<0) then
      J_mat(b_index, h_index) = NaN;
    end if
  end for
end for
BetaSet = cell(length(hh),1);
for index = 1 to length(hh) do
  [~, a] = min(J_mat(:,index));
  BetaSetindex.beta = bbeta(a(1));
  BetaSetindex.h = hh(index);
end for
```

Algorithm A2: Finding β for arbitrary altitude

```
Require: BetaSet and h_in
  return beta
  % BetaSet - set of coefficients generated with CalculateBeta
  % h_in - altitude of the UAV w.r.t.  the final altitude [m]
  betaArray = [BetaSet:];
  hh = [betaArray.h];
  ind1 = find(hh==h_in);
  if ind1 then
    beta = betaArray(ind1).beta;
  else
    step = betaArray(2).h-betaArray(1).h;
    up = floor(h_in) + ceil( (h_in-floor(h_in))/step) * step;
    down = floor(h_in) + floor( (h_in-floor(h_in))/step) * step;
    [ ~ , indUp ] = min( abs( hh-up ) );
    [ ~ , indDown ] = min( abs( hh-down ) );
    weight_up = (h_in-down)/step;
    beta = betaArray(indUp).beta * weight_up + betaArray(indDown).beta
    *(1-weight_up);
  end if
```

References

1. Tognon, M.; Testa, A.; Rossi, E.; Franchi, A. Takeoff and landing on slopes via inclined hovering with a tethered aerial robot. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 1702–1707.
2. Saska, M.; Kasl, Z.; Preucil, Z. Motion planning and control of formations of micro aerial vehicles. *IFAC Proc. Vol.* **2014**, *47*, 1228–1233. [CrossRef]
3. Karaman, S.; Riether, F. Getting started with MIT's Rolling Spider MATLAB Toolbox, An MIT take-home lab for 16.30 Feedback Control Systems, Massachusetts Institute of Technology. 2016. Available online: http://fast.scripts.mit.edu/dronecontrol/wpcontent/uploads/2016/05/GettingStarted.pdf (accessed on 1 April 2020).
4. Mirzaeinia, A.; Hassanalian, M. Minimum-Cost Drone–Nest Matching through the Kuhn–Munkres Algorithm in Smart Cities: Energy Management and Efficiency Enhancement. *Aerospace* **2019**, *6*, 125. [CrossRef]
5. Bayerlein, H.; Kerret, P.D.; Gesbert, D. Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning. In Proceedings of the 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Kalamata, Greece, 25–26 June 2018; pp. 1–5.
6. Baca, T.; Stepan, P.; Spurny, V.; Hert, D.; Penicka, R.; Saska, M.; Thomas, J.; Loianno, G.; Kumar, V. Autonomous landing on a moving vehicle with an unmanned aerial vehicle. *J. Field Robot.* **2019**, *36*, 874–891. [CrossRef]
7. Tan, L.; Wu, J.; Yang, X.; Song, S. Research on optimal landing trajectory planning method between an UAV and a movin vessel. *Appl. Sci.* **2019**, *9*, 3708. [CrossRef]
8. Dono, T.F. *Optimized Landing of Autonomous Unmanned Aerial Vehicle Swarms*; Naval Postgraduate School: Monterey, CA, USA, 2012.
9. Crowther, W.J. Perched landing and takeoff for fixed wind UAVs. In Proceedings of the Unmanned Vehicles for Aerial, Ground and Naval Military Operations RTO AVT Symposium, Ankara, Turkey, 9–13 October 2000; pp. 19-1–19-10.
10. Fang, X.; Wan, N.; Jafarnejadsani, H.; Sun, D.; Holzapfel, F.; Hovakimyan, N. Emergency Landing Trajectory Optimization for Fixed-Wing UAV under Engine Failure. In Proceedings of the AIAA Scitech Forum 2019, San Diego, CA, USA, 7–11 January 2019.
11. Zhang, H.; Liu, C.; Xu, G. Take-off trajectory optimization of vertical take off and landing UAV. In Proceedings of the 36th Chinese Control Conference (CCC), Dalian, China, 26–28 July 2017. [CrossRef]
12. Saeed, A.S.; Younes, A.B.; Cai, C.; Cai, G. A survey of hybrid Unmanned Aerial Vehicles. *Prog. Aerosp. Sci.* **2018**, *98*, 91–105. [CrossRef]

13. Liu, H. Multivariable Control of a Rolling Spider Drone. Master's Thesis, University of Rhode Island, Kingston, RI, USA. Available online: https://digitalcommons.uri.edu/theses/1064 (accessed on 12 March 2019).
14. Ryll, M.; Bicego, D.; Franchi, A. Modeling and Control of FAST-Hex: A Fully-Actuated by Synchronized-Tilting Hexarotor. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Dajeon, Korea, 9–14 October 2016; pp. 1689–1694.
15. Horla, D.; Owczarkowski, A. Robust LQR with actuator failure control strategies for 4DoF model of unmanned bicycle robot stabilised by inertial wheel. In Proceedings of the International Conference on Industrial Engineering and Systems Management, Seville, Spain, 21–23 October 2015; pp. 998–1003.
16. Horla, D. *Metody obliczeniowe optymalizacji w zadaniach*, 2nd ed.; Wydawnictwo Politechniki Poznanskiej: Poznan, Poland, 2016. (In Polish)
17. Zhang, J.; Mcinnes, C.R. Using instability to reconfigure smart structures in a spring-mass model. *Mech. Syst. Signal Process.* **2017**, *91*, 81–92. [CrossRef]
18. Athans, M.; Falb, P.L. *Optimal Control: An Introduction to the Theory and Its Applications*; McGraw-Hill: New York, NY, USA, 1966.
19. Goslinski, J.; Giernacki, W.; Krolikowski, A. A Nonlinear Filter for Efficient Attitude Estimation of Unmanned Aerial Vehicle (UAV). *J. Intell. Robot. Syst.* **2019**, *95*, 1079–1095. [CrossRef]
20. Giernacki, W.; Horla, D.; Sadalla, T. Mathematical Models Database (MMD ver. 1.0) Non-commercial proposal for researchers. In Proceedings of the 21st International Conference on Methods and Models in Automation and Robotics (MMAR), Miedzyzdroje, Poland, 24–27 August 2016; pp. 555–558.
21. Zhang, J.; Mcinnes, C.R. Reconfiguring smart structures using approximate heteroclinic connections. *Smart Mat. Struct.* **2015**, *24*, 105034. [CrossRef]
22. Hassanalian, M.; Abdelkefi, A. Classifications, applications, and design challenges of drones: A review. *Prog. Aerosp. Sci.* **2017**, *91*, 99–131. [CrossRef]