Hindawi
International Journal of Aerospace Engineering
Volume 2020, Article ID 1372052, 13 pages
https://doi.org/10.1155/2020/1372052

Research Article

Helicopter Autorotation Trajectory Planning Method Using Functional Tensor-Train-Based Dynamic Programming Algorithms

Yunjie Wang, Chen Jiang, Shuai Deng, and Haowen Wang

School of Aerospace, Tsinghua University, Beijing 100084, China

Correspondence should be addressed to Haowen Wang; bobwang@mail.tsinghua.edu.cn

Received 11 January 2020; Accepted 7 May 2020; Published 5 July 2020

Academic Editor: Luis E. González-Jiménez

Helicopter autorotation trajectory planning problems have been dealt within computationally expensive optimal control algorithms. This paper presents an efficient helicopter autorotation trajectory planning method, using functional tensor-train-(FT-) based dynamic programming (DP) algorithms. The autorotation trajectory planning method is shown real-time feasible, which involves general helicopter autorotation dynamics at the same time. To validate the dynamic feasibility of the trajectories, a trajectory-tracking controller using active disturbance rejection control (ADRC) is designed to ensure a helicopter model tracks the trajectories. Finally, a helicopter autorotation simulation with a six-degree-of-freedom high-fidelity multibody-based helicopter model is demonstrated for validation.

1. Introduction

Autorotation is a primary measure for helicopters, whether for manned or unmanned helicopters, to land safely after engine power failure. In autorotation, the main rotor is driven by the upward airflow through the rotor, making the flight similar to a gliding fixed-wing aircraft. To achieve a safe autorotation landing, unique control strategies are needed [1]. Generally speaking, the collective pitch should be carefully handled to maintain a sufficient and steady rotor speed before reaching the ground.

Studies on means of achieving autorotation have been focused on solving optimal control problems [2–4]. Traditionally, simplified helicopter dynamic models are used, including 2-D point-mass models [4–6], three-degree-of-freedom rigid-body models [7, 8], and low-order six-degree-of-freedom rigid-body models [3]. With the dynamic model formulated, optimal autorotation problems are solved by numerical methods. Gradient algorithms, such as sequential gradient-restoration algorithms, are used in [5, 7, 9]. Other methods, such as direct methods, discretize the problem first and turn it into a nonlinear programming problem. Then, the nonlinear programming problem can be solved using algorithms such as sequential quadratic programming (SQP), as demonstrated in [6, 10, 11]. In general, these algorithms are off-line, computationally expensive for current computing power, which is for now unrealistic for on-line usages. Furthermore, there is a lack of validations either through high-fidelity simulations or experiments in the above studies.

Recently, Taamallah [12] proposed the first real-time feasible, model-based trajectory planning method and designed a model-based trajectory-tracking controller to ensure the helicopter tracks the trajectories. In specific, they employed optimal planning based on differential flatness, assuming a helicopter as a rigid body. The trajectory planning problem is solved, regarding all the rotor forces and moments as plant inputs of the rigid body. Taamallah's work provides novel directions for online trajectory planning of autorotation problems. However, as a consequence, forces and moments are simplified and decoupled with helicopter dynamics in such methods. It is necessary to mention that, in autorotation procedures, the main rotor's ability to generate forces is highly restrained by the states of helicopters, especially by the main rotor's rotating speed and the inflow state of the rotor [13]. The rotor speed is the crucial factor that

determines whether a trajectory leads to a safe autorotation landing. Thus, the dynamics of autorotation is of great significance for trajectory planning and should be involved explicitly.

In this paper, we present a real-time feasible autorotation trajectory planning method using functional tensor-train- (FT-) based dynamic programming (DP) algorithms, which ensures that general autorotation dynamics is satisfied along the trajectory. For validation of the dynamic feasibility of the trajectories, we also present a trajectory-tracking controller based on active disturbance rejection control (ADRC) to make a helicopter model track the trajectories. The validation of the trajectories using the controller is then implemented on a six-degree-of-freedom, high-fidelity, multibody-based helicopter simulation model.

The trajectory planning method using functional tensor-train- (FT-) based dynamic programming (DP) algorithms will be presented in Section 2. The ADRC-based trajectory-tracking controller is described in Section 3. In Section 4, we describe the six-degree-of-freedom, high-fidelity, multibody helicopter simulation model using the Tsinghua Rotorcraft Utility Simulation Tool (TRUST). In Section 5, autorotation trajectories are demonstrated with various initial conditions, and simulation results for validation are demonstrated and discussed. Finally, conclusions are presented in Section 6.

2. Autorotation Trajectory Planning Using FT-Based DP Algorithms

As mentioned in Section 1, traditional trajectory planning methods for autorotation have been dealt within computationally expensive off-line algorithms. In 2017, the first real-time feasible trajectory planning method for autorotation was demonstrated in [12], which is based on differential flatness of the rigid-body dynamics. Such methods simplify forces as direct inputs, thus leaving the helicopter autorotation dynamics not included during trajectory planning procedures. In this section, we introduce a real-time feasible trajectory planning method, using functional tensor-train- (FT-) [14] based dynamic programming (DP) algorithms, which guarantees a strict satisfaction of helicopter dynamics along the trajectory.

2.1. FT-Based DP Algorithms. Functional tensor-train-based (FT-based) dynamic programming (DP) algorithms are newly proposed algorithms for solving high-dimensional stochastic optimal control (SOC) problems, which are mainly discounted-cost infinite-horizon Markov decision process (MDP) problems. Here, we give a brief review of the FT-based DP algorithms, and details can be found in [15, 16].

Consider a system described by stochastic differential equations (SDE) as follows:

$$\mathbf{x}(t) = \mathbf{b}(t, \mathbf{x}(t), \mathbf{u}(t))dt + \boldsymbol{\sigma}(\mathbf{x}(t))d\mathbf{w}(t), \tag{1}$$

where $\mathbf{x}(t) \in \mathbf{R}^n$ is the state vector, $\mathbf{u}(t) \in \mathbf{R}^m$ is the control input, $\mathbf{w}(t)$ is a vector of independent unit Wiener processes, and $\mathbf{b}(\cdot)$ and $\boldsymbol{\sigma}(\cdot)$ are generally nonlinear functions. The cost functional is defined as

$$J(\mathbf{x}(t_0), \mathbf{u}) = E\left[\int_{t_0}^{\tau} e^{-\beta s} g(s, \mathbf{x}(s), \mathbf{u}(s))ds + \chi_{\tau<\infty} e^{-\beta\tau} \psi_J(\tau, \mathbf{x}(\tau))\right], \tag{2}$$

where β is a discount factor, $\chi_{\tau<\infty}$ is the indicator function of the state boundary, and $g(\cdot)$ and $\psi_J(\cdot)$ are stage cost function and terminal cost function. The SOC problem is to find a control $\mathbf{u}(t)$ within a specified set on the time interval $[t_0, \tau]$, such that the cost $J(\mathbf{x}(t_0), \mathbf{u})$ is minimized.

Next, using the MCA method [17], continuous SOC problems are discretized into the discrete Markov decision processes (MDPs). The discretized problem is turned into searching for a value function that satisfies the following recursive equation:

$$v(\mathbf{z}) = \min_{\mu}\left[g(\mathbf{z}, \mathbf{u}) + \gamma \sum_{\mathbf{z}'} p\left(\mathbf{z}, \mathbf{z}' \middle| \mathbf{u}\right) v\left(\mathbf{z}'\right)\right], \tag{3}$$

where $v(\mathbf{z})$ is the optimal discretized value function and $p(\mathbf{z}, \mathbf{z}' | \mathbf{u})$ is the transition probability function.

Then, a discounted-cost infinite-horizon MDP is formulated and can be solved by the FT-based DP algorithms, which are FT-based value iteration algorithm, FT-based policy iteration algorithm, and FT-based multilevel algorithm, respectively.

For the traditional discrete-state Markov decision processes, computational requirements grow exponentially with dimensionality. For example, if a MDP has 8 dimensions, and each dimension has a discretization of 10 points, such a problem involves a search space of 10^8 points. In order to mitigate such curse of dimensionality, FT-based DP algorithms use low-rank-functions, namely functional tensor-train, to represent value functions. The basic idea of function-train (FT) is to make a continuous analogue [14] of the tensor-train decomposition [18]. To be specific, it is a continuous version of tensor-train cross-approximation (TT-cross-approximation) [19], with the formulation as follows:

$$f(x_1, \cdots, x_d) = \prod_{i=1}^{d} \mathbf{F}_i(x_i), \tag{4}$$

where $f(x_1, \cdots, x_d)$ is a d-dimensional multivariable function. $\mathbf{F}_i(x_i)$ is a set of univariate functions, which are also called *cores*:

$$\mathbf{F}_i(x_i) = \begin{bmatrix} f^{(i)}_{1,1}(x_i) & \cdots & f^{(i)}_{1,r_i}(x_i) \\ \vdots & \ddots & \vdots \\ f^{(i)}_{r_{i-1},r_i}(x_i) & \cdots & f^{(k)}_{r_{i-1},r_i}(x_i) \end{bmatrix}, \tag{5}$$

where r_i are the FT ranks evaluated by a continuous version of TT-rounding [14] and $f_{i,j}^{(k)}$ are univariate hat functions.

With methods described above, the exponentially growing computational complexity of $O(n^d)$ for a typical dynamic programming problem is compressed to a polynomially growing complexity of

$$O\left(dnr^2\kappa\left(n_{\text{op}} + d^2nr^2\right) + dnr^3\right), \tag{6}$$

where κ is the operations during value evaluation, n_{op} is the step of operations within each Bellman equation step.

To summarize, this method formulates a SOC problem as a dynamic programming problem and implements a continuous tensor decomposition method to compress such a problem, resulting in significant improvements of computing speed and storage savings. Such method has been shown to be able to work in real-time [15].

2.2. Formulating Autorotation Trajectory Planning Problems within FT-Based DP Algorithms. In this part, we describe a general method for solving autorotation trajectory planning problems in the framework of FT-based DP algorithms. First of all, we need to describe the dynamics of a helicopter in a SOC form, as equation (1) shows.

2.2.1. Helicopter Dynamic Model Formulation. The helicopter dynamic function in the SOC equations is given by a nonlinear three-degree-of-freedom rigid-body helicopter autorotation dynamic model. Note that in this paper, two different helicopter dynamic models are described. The first one is the nonlinear three-degree-of-freedom rigid-body helicopter dynamic model, which is used in this section as the system dynamics in equation (1). The second one is the six-degree-of-freedom, high-fidelity, multibody-based helicopter simulation model, which will be used as a validation model in simulation and will be described in Section 5. The three-degree-of-freedom dynamic model is chosen here for computation considerations, and such model is capable of predicting steady collective pitch manipulations and rotor power consumption [13].

States variables for the nonlinear three-degree-of-freedom rigid-body helicopter dynamic model are chosen as

$$\mathbf{s} = (z, v_x, v_z, \theta, q, \Omega), \tag{7}$$

which include height z, horizontal velocity v_x, rate of descent v_z, pitch angle θ, pitch rate q, and rotating speed of the main rotor Ω. The control variables are collective pitch θ_0 and longitudinal cyclic pitch θ_{1s}. The dynamic model equations are formulated as follows [20]:

$$\begin{cases} \dot{z} = v_z \\ \dot{v}_x = -\dfrac{1}{m}\left(T_{\text{B}}\sin\theta + H_{\text{B}}\cos\theta + D_{\text{F}}\cos\alpha_{\text{F}}\right) \\ \dot{v}_z = -\dfrac{1}{m}\left(T_{\text{B}}\cos\theta - H_{\text{B}}\sin\theta + D_{\text{F}}\sin\alpha_{\text{F}}\right) + g \\ \dot{\theta} = q \\ \dot{q} = \dfrac{1}{I_{FY}}\left(M_Y - T_{\text{B}}\cdot l_{\text{R}} + H_{\text{B}}\cdot h_{\text{R}} - L_{\text{H}}\cos(\theta-\gamma)\cdot l_{\text{H}}\right) \\ \dot{\Omega} = -\dfrac{1}{I_{\text{R}}}Q, \end{cases} \tag{8}$$

where T_{B} and H_{B} are rotor forces resolved in the body axes. Note that all the axes are defined in accordance with [13]. M_Y and Q are pitch moment and rotor torque, respectively. D_{F} and L_{H} are the drag of the fuselage and the lift of the horizontal tail resolved in the body axes. α_{F} and γ are the angle of attack of the fuselage and the flight-path angle. h_{R} and l_{R} are vertical and horizontal distance of rotor hub to the center of gravity. l_{H} is the horizontal distance of the horizon tail from the center of gravity. Details of forces generated by the fuselage and the horizontal tail can be seen in [20].

Next, the rotor performance during autorotation is given by the following procedures. The induced velocity v is calculated by [9]:

$$v = \kappa v_{\text{h}} f_{\text{I}} f_{\text{G}}, \tag{9}$$

where κ is the empirical correction factor of nonuniform inflow and v_{h} is the induced velocity at hover. f_{I} is the induced velocity parameter which has included the influence of the vortex-ring state, given by [9]:

$$f_{\text{I}} = \begin{cases} 1.0/\sqrt{\bar{v}_{xb}{}^2 + \left(-\bar{v}_{zb} + f_{\text{I}}\right)^2}, \text{if}\left(-2\bar{v}_{zb} + 3\right)^2 + \bar{v}_{xb}{}^2 \geq 1.0 \\ -\bar{v}_{zb}\left(0.373\bar{v}_{zb}{}^2 + 0.598\bar{v}_{xb}{}^2 - 1.991\right), \text{otherwise}, \end{cases} \tag{10}$$

where $\bar{v}_{xb}$ and $\bar{v}_{zb}$ are normalized velocities resolved in body axes:

$$\begin{bmatrix} \bar{v}_{xb} \\ \bar{v}_{zb} \end{bmatrix} = \frac{1}{v_{\text{h}}}\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} v_x \\ v_z \end{bmatrix}. \tag{11}$$

The ground effect factor f_{G} is expressed as [13]

$$f_{\text{G}} = 1 - \frac{1}{1 - (R/4h)^2}, \tag{12}$$

where $h = z + z_0$ and z_0 is the rotor height when the aircraft is on the ground. Note that we ignore the influence of horizontal speed on the ground effect, and such assumption is reasonable for a typical autorotation procedure.

The rotor thrust coefficient C_T is expressed as [13]

$$C_T = \frac{1}{2}a\sigma\left[\left(\frac{1}{3}+\frac{1}{2}\bar{v}_{xb}{}^2\right)\left(\theta_0-\frac{3}{4}\theta_{tw}\right)B^3 + \frac{1}{4}\left(1+\bar{v}_{xb}{}^2\right)B^4-\frac{1}{2}\lambda\left(1+\frac{1}{2}\bar{v}_{xb}{}^2\right)B^2\right], \tag{13}$$

where a is the blade lift-curve slope, σ is the rotor solidity ratio, θ_{tw} is the blade twist, and B is the blade tip-loss factor.

The rotor drag force coefficient C_H is made up of the rotor profile drag C_{H0} and the induced drag C_{Hi}. The rotor pitch moment coefficient is C_{MY}. C_H and C_{MY} are expressed as follows:

$$\begin{cases} C_H = C_{H0} + C_{Hi} \\ C_{H0} = \dfrac{c_d\sigma}{8}\left(3\bar{v}_{xb}+1.98\bar{v}_{xb}{}^{2.7}\right) \\ C_{Hi} = \dfrac{a\sigma}{2}\left[\left(-\dfrac{1}{3}\beta_c+\dfrac{1}{2}\bar{v}_{xb}\lambda\right)\theta_0-\dfrac{1}{8}\theta_{tw}\bar{v}_{xb}\lambda+\left(-\dfrac{1}{4}\bar{v}_{xb}\beta_{1c}+\dfrac{1}{4}\lambda\right)\theta_{1s}+\dfrac{3}{4}\lambda\beta_{1c}+\dfrac{1}{6}\beta_0\beta_{1c}+\dfrac{1}{4}\left(\beta_0{}^2+\beta_{1c}{}^2\right)\right] \\ C_{MY} = -\dfrac{1}{2}\varepsilon n_b\Omega^2\beta_{1c}S_b, \end{cases} \tag{14}$$

where c_d is the blade's mean profile drag coefficient, ε is the flap offset, n_b is the number of blades, and S_b is the blade's first moment about the flap hinge. β_0 is the coning angle of rotor disk, and β_{1c} is the first harmonic cyclic flap, given by

$$\begin{cases} \beta_0 = \gamma\left[\dfrac{1}{8}\left(\theta_0-\dfrac{3}{4}\theta_{\text{tw}}\right)\left(1+\bar{v}_{xb}{}^2\right)B^4+\dfrac{\theta_{tw}}{10}\left(1+\dfrac{5}{6}\bar{v}_{xb}{}^2\right)B^5-\dfrac{1}{6}(\lambda-\bar{v}_{xb}\theta_{1s})B^3-\dfrac{3}{2}\dfrac{g}{\Omega^2R}\right] \\ \beta_{1c} = \dfrac{-8/3[\theta_0-3/4(\lambda-\bar{v}_{xb}\theta_{1s})]}{1-1/2\bar{v}_{xb}{}^2}-\theta_{1s}, \end{cases} \tag{15}$$

where λ is the normalized inflow velocity, expressed as

$$\lambda = v/\Omega R - \bar{v}_{zb}. \tag{16}$$

The power coefficient required by the main rotor C_P is equal to the rotor torque coefficient C_Q, which is made up of the rotor induced power, the rotor profile power, the rotor parasite power, and the rotor climb power. A general expression is given by [13]

$$C_P = \kappa C_T\frac{v}{\Omega R}+\frac{1}{8}\sigma c_d\left(1+4.6\bar{v}_{xb}{}^2\right)+\frac{1}{2}\bar{v}_{xb}{}^3\frac{f}{\pi R^2}-\bar{v}_{zb}C_T, \tag{17}$$

where f is the fuselage equivalent parasite drag area.

2.2.2. Formulation of SOC Problem for Autorotation Trajectory Planning. Following the helicopter autorotation dynamic model described above, and for numerical considerations [5], the stochastic differential equations are obtained by normalizing equations of (8). The normalized states and inputs are as follows:

$$\begin{cases} x_1 = \dfrac{1}{10R}z, x_2 = \dfrac{100}{\Omega_0R}v_x, x_3 = \dfrac{100}{\Omega_0R}v_z, x_4 = 10\theta, x_5 = \dfrac{1000}{\Omega_0}q, x_6 = \dfrac{1}{\Omega_0}\Omega \\ u_1 = 10\theta_0, u_2 = 10\theta_{1s}. \end{cases} \tag{18}$$

In addition, the Jacobian matrix of control inputs $[\partial\dot{x}_i/\partial u_j], i = 1, \cdots, 6, j = 1, 2$ is calculated by numerical difference method for any state $\mathbf{x}$ and input $\mathbf{u}$. The diffusion function $\boldsymbol{\sigma}(\cdot)$ is set to 10^{-9} for each term.

Next, the cost functions are designed. The stage cost $g(\mathbf{x}, \mathbf{u})$ is expressed by

$$g(\mathbf{x},\mathbf{u}) = W_1x_1{}^2+W_2x_2{}^2+W_3x_3{}^2+W_4x_4{}^2+W_6(1-x_6)^2+W_7u_1{}^2+W_8u_2{}^2. \tag{19}$$

The terminal cost $\psi_J(\mathbf{x})$ is given by setting an absorbing region [15], which encourages a safe landing, as shown below:

$$\psi_J(\mathbf{x}) = \begin{cases} 0 & \text{if safely landed} \\ W_{\mathrm{T1}}{x_1}^2 + W_{\mathrm{T2}}\left(x_2 - x_{2\mathrm{glide}}\right)^2 + W_{\mathrm{T3}}(x_3)^2 + W_{\mathrm{T4}}{x_4}^2 + W_{\mathrm{T5}}{x_5}^2 & \text{otherwise,} \end{cases} \tag{20}$$

where $W_{\cdot}$ and $W_{\mathrm{T}\cdot}$ denote weighting factors of stage cost and terminal cost, respectively. We add cost functions to control inputs in order to avoid fierce manipulations. $x_{2\mathrm{glide}}$ is the average horizontal speed, estimated by empirical flight test results and numerical simulation results. A safe autorotation landing is defined by

$$\begin{aligned} v_z(\tau) \le v_{z\ \max}, |v_x(\tau)| \le v_{x\ \max}, |\theta(\tau)| \\ \le \theta_{\max}, |q(\tau)| \le q_{\max}, z = 0. \end{aligned} \tag{21}$$

The design of cost functions is shown to be important for such autorotation problems, and parameters are needed to be adjusted by numerical experiments.

2.2.3. Trajectory Generation Using FT-Based Algorithms. After the problem is formulated in the form of a SOC problem in Section 2.2.2, solutions are then obtained using FT-based algorithms.

Although FT-based algorithms are proposed for stochastic optimal control problems, we slightly alter such algorithms to generate a trajectory, rather than to obtain control inputs directly. The reason is that even though the nonlinear dynamics of helicopter described in Section 2.2 gives nice results in terms of predicting steady collective pitch manipulations and rotor power consumption, such model is not able to make good predictions of the helicopter dynamic response [21]. Therefore, the control inputs generated by the SOC controller are not adopted, and we make use of the trajectory instead. As mentioned before, the results of rotor power and collective pitch along the trajectory can be regarded reasonable.

We employ the FT-based one-way multigrid algorithm proposed in [15, 22]. For each discretization level, $N_{\mathrm{iter}} = 100$ steps of FT-based policy iterations are applied. The trajectory of normalized states $\mathbf{x}$ is obtained by an integration procedure of given initial conditions to the controller:

$$\{\mathbf{x}(t_i)\} = \left\{ \int_{t_{i-1}}^{t_i} \mathbf{b}(\mathbf{x}(t_{i-1}), \mathbf{u}(\mathbf{x}_{i-1})) \right\}, t_i \in [t_0, \tau], \mathbf{x}(t_0) = \mathbf{x}_0. \tag{22}$$

In specific, a fourth-order order Runge-Kutta method is implemented. Thus, preliminary trajectories are obtained by making $\mathbf{x}$ unnormalized by equation (18). Because the trajectories are generated by integrations based on the dynamic equations of the helicopter, they satisfy the specified helicopter dynamics by nature.

However, although such trajectories are generated by solving dynamic equations, the trajectories still need to be smoothed. There are two main reasons. The first reason is that for real-time realizability, the integration time step (for example, 0.05 s in our study) is not small enough for a controller to generate sufficient differential information. The second reason is that although we have cost functions on control inputs, the control values still result in discontinuous variations, making the trajectory not smooth enough for our controller. Thus, we apply a fast interpolation method, i.e., cubic Hermite interpolation, to obtain smooth trajectories.

3. Autorotation Trajectory-Tracking Controller Based on ADRC

In order to show the practicability of trajectories generated by FT-based DP methods, we demonstrate a trajectory-tracking controller based on active disturbance rejection control (ADRC) and use the controller to make a high-fidelity helicopter model track the trajectories. Tracking of position and velocity is vital for a successful autorotation landing, and the dynamics during autorotation may be more complicated than the formulations described in Section 2.2. Besides, there also exist unexpected disturbances during real flights. Thus, we use active disturbance rejection control (ADRC) methods to design the tracking controller. Active disturbance rejection control (ADRC) is proposed by Han [23], and the ADRC controller is capable of estimating inner modelling errors or outer disturbances, thus making compensation accordingly.

In specific, we implement the ADRC-based trajectory-tracking controller described in [24], and such controller has been successfully validated through flight tests [24, 25]. The structure of the controller is shown in Figure 1: where $\mathbf{x}_{\mathrm{traj}}$ and $\mathbf{v}_{\mathrm{traj}}$ are position vectors and velocity vectors of the trajectory and $\mathbf{u}_{\mathrm{r}}$ represents the reference velocities that needed to be tracked by the inner loop controller.

The order of the ADRC controller differs with different channels. For the autorotation application, we use a 3rd-order ADRC controller for forward, vertical, lateral channels, and a PI controller for yaw channel. In general, an ADRC controller is made up of a tracking differentiator (TD), an extended state observer (ESO), and a nonlinear state error feedback (NLSEF). Disturbances or modelling errors of the system are observed by the ESO, and then NLSEF is utilized to restrain them. A typical architecture of an ADRC controller is shown in Figure 2.

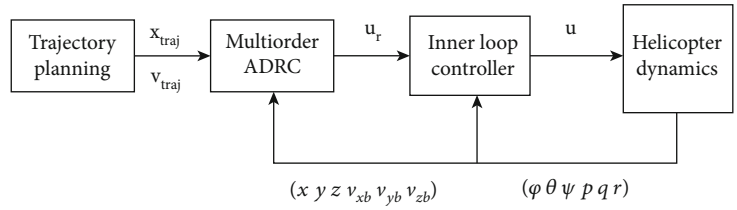

Figure 1: Trajectory-tracking controller structure.

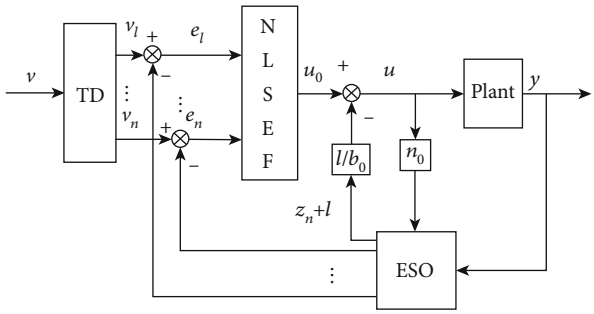

Figure 2: ADRC architecture.

Where v is the input signal, b_0 is the input gain factor, $z_{\cdot}$ is the output of the ESO. For brevity, only the 3rd-order ADRC controller is presented here. The TD of a 3rd-order is formulated as [25]

$$\begin{cases} \text{fh}_1 = \text{fhan}(v_1 - v_0, v_2, r, h_0) \\ v_1 = v_1 + h_0 \cdot v_2 \\ v_2 = v_2 + h_0 \cdot \text{fh}_1 \\ \text{fh}_2 = \text{fhan}(v_2, v_3, r, h_0) \\ v_3 = v_3 + h_0 \cdot \text{fh}_2, \end{cases} \tag{23}$$

where $\text{fhan}(\cdot)$ is an optimal control function defined in [23] and r and h_0 are controller parameters. The 3rd-order ESO is formulated as

$$\begin{cases} e = z_1 - y \\ z_1 = z_1 + h(z_2 - \beta_{01} e) \\ z_2 = z_2 + h(z_3 - \beta_{02} \cdot \text{fal}(e, 0.5, \delta)) \\ z_3 = z_3 + h(z_4 - \beta_{03} \cdot \text{fal}(e, 0.25, \delta) + |b_0| u) \\ z_4 = z_4 + h(-\beta_{04} \cdot \text{fal}(e, 0.125, \delta)), \end{cases} \tag{24}$$

where h is the integration step of the system, $\beta_{\cdot}$ is the observer gain, and function $\text{fal}(\cdot)$ is defined in [23]. The 3rd-order NLSEF is formulated as

$$\begin{cases} e_1 = v_1 - z_1 \\ e_2 = v_2 - z_2 \\ e_3 = v_3 - z_3 \\ u_0 = \beta_1 \text{fal}(e_1, \alpha_1, \delta) + \beta_2 \text{fal}(e_2, \alpha_2, \delta) + \beta_3 \text{fal}(e_3, \alpha_3, \delta) \\ u = u_0 - z_4 / |b_0|. \end{cases} \tag{25}$$

Note that because in this paper we focus on the longitudinal performance of the controller, the yaw controller can be regarded as a yaw stabilizer.

The input signal of the ADRC is processed by the following procedure:

$$\begin{bmatrix} v_{xb\_\text{ref}} \\ v_{yb\_\text{ref}} \\ v_{zb\_\text{ref}} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & \cos\phi & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} v_{x\_\text{traj}} - v_x \\ v_{y\_\text{traj}} - v_y \\ K_z(z_{\text{traj}} - z) + v_{z\_\text{traj}} - v_z \end{bmatrix}, \tag{26}$$

where x_{traj}, z_{traj}, v_{x_traj}, and v_{z_traj} are positions and velocities along the trajectory, K_z is the vertical factor, and $(v_{xb\_\text{ref}}, v_{xb\_\text{ref}}, v_{xb\_\text{ref}})$ is the input signal of the ADRC controller.

The inner loop controller is a PI controller with damping feedbacks, and such controller is proven to be working well with ADRC in trajectory-tracking applications [25].

Following the above process, a trajectory controller is obtained. It is needed to mention that when the helicopter is near the ground, the controller is reset to make the helicopter descent slowly. Such techniques to ensure a stable landing is also reported by [12].

4. The Tsinghua Rotorcraft Utility Simulation Tools (TRUST)

This section briefly introduces the Tsinghua Rotorcraft Utility Simulation Tools (TRUST), with which we model the S-58 helicopter and validate the trajectories using the ADRC controller. Tsinghua Rotorcraft Utility Simulation Tools (TRUST) stems from [26], which is based on the framework of multibody dynamics. The TRUST simulator is able to simulate many kinds of rotorcrafts, including regular helicopters, tandem helicopters, and compound helicopters, with articulated or flexible rotor hubs. Regular helicopter models consist of main rotor dynamics, tail rotor dynamics, rigid-body fuselage dynamics, horizontal tails, and vertical tails. Such models have proven to be of good agreement with various test data [26].

4.1. Model Description. Considering both fidelity and simulation feasibility, we adopt the following modelling methods.

4.1.1. Main Rotor. Main rotor is modelled as an articulated rotor with rigid blades of certain twist. Each blade is attached to the hub through pitch, lag, and flap hinges. The inflow is modelled as the three-state Pitt-Peters dynamic inflow [27]. Blade lift and drag forces are calculated using the blade

Table 1: S-58 specifications.

Parameter	Value used	Parameter	Value used
Rotor radius, m	8.53	Fuselage inertia I_{yy}, kg m^2	48796
Blade chord, m	0.417	Fuselage inertia I_{xx}, kg m^2	16265
Blade twist, deg	-8	Fuselage inertia I_{zz}, kg m^2	48796
Number of blades	4	HT area, m^2	1.15
Flap hinge offset, m	0.3048	Lift-curve slope of HT	3.73
Rotating inertia of the MR, kg m^2	7101.2	Rotor hub height above ground, m	4.36
Tilting angle, deg	0	Fuselage flat plate area, m^2	3.55

element theory, and lift-curve slope is obtained from a nonlinear tabulate of angle of attack and relative airflow speed.

In addition, because the main rotor may traverse from powered lift state into vortex-ring state (VRS) during autorotation, a modification [28] of vortex-ring state is applied to the three-state Pitt-Peters model.

Due to the multibody dynamic framework, each blade's motion is solved individually by the Newton/Euler equations. Thus, flap angles of blades are not assumed to be small, nor the blades motions are treated as a consequence of a periodical disk, which involves inevitable accuracy loss for blade motion responses [29].

4.1.2. Tail Rotor. Tail rotor is modelled similar to the main rotor, except that the rotor hub is rigid and no VRS correction is applied.

4.1.3. Fuselage. Fuselage is regarded as a rigid body, with linear aerodynamic lift and drag coefficients.

4.1.4. Horizontal and Vertical Tails. The aerodynamic forces of horizontal and vertical tails are calculated by flat plate models.

For the purpose of validating autorotation trajectory planning and tracking, the nonlinear helicopter model described above is capable of predicting dynamic responses of the helicopter [28, 30].

4.2. Dynamics of TRUST Model. As mentioned in Section 2, there are two different helicopter dynamic models used in our study. The nonlinear three-degree-of-freedom rigid-body helicopter dynamics is capable of calculating steady collective pitch manipulations, rotor forces, and power, but not sufficient for dynamic response predictions. Due to the fact that the rotor forces in such models are essentially zero-order processes, dynamic responses of the rotor cannot be well described.

In the TRUST helicopter model, a more sophisticated dynamic inflow model is involved, and rotor blade motions are accurately described in the multibody framework. Blade lift and drag forces are from nonlinear tabulates of experiment data. Furthermore, helicopter dynamics is well known for coupling effects between different channels [31]. Hence, for validation purposes, a high-order, six-degree-of-freedom, helicopter dynamic model should be considered.

5. Numerical Experiments

In the first part, we show the trajectory planning results of various autorotation initial conditions based on the Sikorsky S-58. In the second part, to validate the dynamic feasibility of the trajectory planning results, we implement the ADRC controller and validate the trajectory planner by making a 6-DOF nonlinear high-fidelity S-58 model track the trajectory.

The Sikorsky S-58 is a single-engined helicopter equipped with a 4-bladed articulated rotor with a radius of 8.535 m. The take-off weight used in our research is 4500 kg. Detailed parameters can be found in [32, 33], and main parameters among which can be found in Table 1.

5.1. Autorotation Trajectory Planning Results. The parameters of the cost functions in equations (19) and (20) are set as follows:

$$W_1 = 50.0,\ W_2 = 12.5,\ W_3 = 5.0,\ W_4 = 1.0,\ W_5 = 0.5,\ W_6 = 10.0$$
$$W_7 = 1.0,\ W_8 = 1.0$$
$$W_{T1} = 400.0,\ W_{T2} = 0.25,\ W_{T3} = 0.25,\ W_{T4} = 0.2,\ W_{T5} = 0.03. \tag{27}$$

Note that costs for control inputs are added to avoid extreme manipulations. The empirical gliding speed is set as $x_{2\text{glide}} = 5.0$. The definition of a successful autorotation landing is defined as

$$v_z \le 2\text{m/s},\ |v_x| \le 3\text{m/s},\ |\theta| \le 12 \text{ deg}. \tag{28}$$

In addition, we also implement a start cost function before the value iterations. A start cost function is used to initialize the global value function before the iterations begin. The start cost function can be expressed as follows:

$$J_{start} = \begin{cases} \psi_J & \text{if } |z| > h_p \\ \psi_J + \left(\dfrac{|x_1|}{h_p} - 1\right)^2 \left[15{x_2}^2 + 15{x_3}^2 + 20{x_4}^2\right] & \text{otherwise,} \end{cases} \tag{29}$$

where h_p is set to 4.0 m in this case. We use the one-way method of $N = 20, 40$ points for each dimension and set

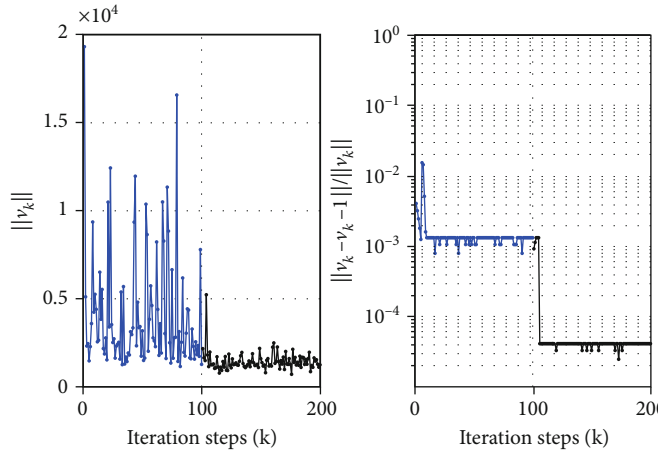

Figure 3: FT-based algorithms value iteration plots. The blue line denotes values of $N = 20$, and the black line denotes values of $N = 40$.

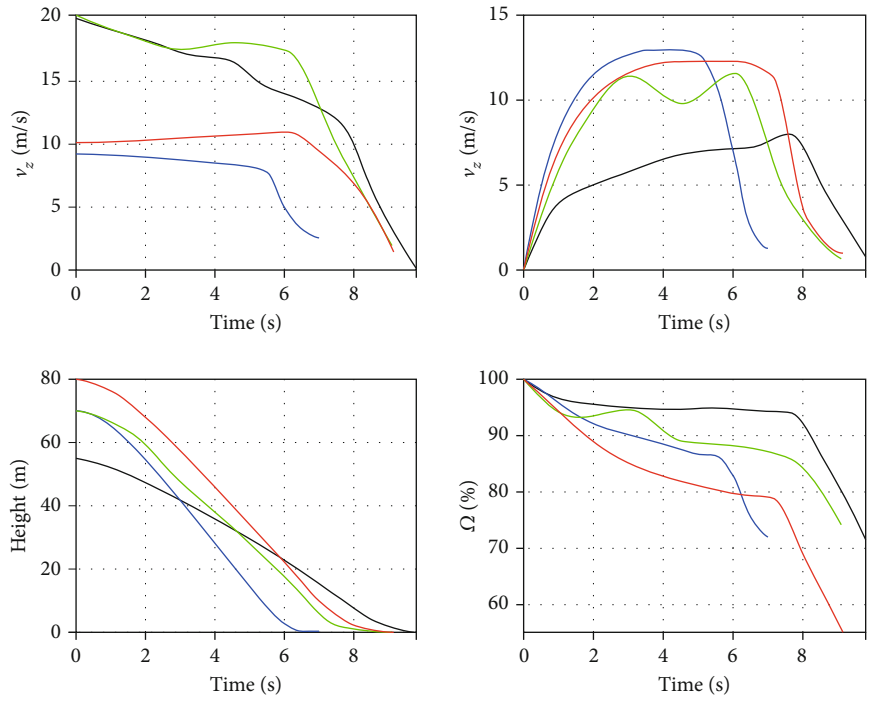

Figure 4: Autorotation trajectories. Black lines for Case 1, blue lines for Case 2, green lines for Case 3, and red lines for Case 4.

the max rank approximation of the core functions to $r_{\max} = 15$. Using the one-way method, the initial value function of $N = 40$ is obtained from the solutions of $N = 20$. Convergence is plotted in Figure 3.

The left panel shows when $N = 40$, the value function converges to a certain value around 2000 quickly. The relative error of the value function between different iterations is plotted in the right panel. It is shown that for the discretization of $N = 40$, the relative error is small for a value of about 3×10^{-5}, which is near the converging threshold of 10^{-5}.

Various autorotation cases using the trajectory planner are tested, with different initial height and horizontal speed:

Case 1. $\mathbf{s_0} = (-55\text{m}, 20\text{m/s}, 0\text{m/s}, -3.7\text{ deg}, 0\text{ deg/s}, 235\text{RPM})$.

Case 2. $\mathbf{s_0} = (-70\text{m}, 10\text{m/s}, 0\text{m/s}, -2.4\text{ deg}, 0\text{ deg/s}, 235\text{RPM})$.

Case 3. $\mathbf{s_0} = (-70\text{m}, 20\text{m/s}, 0\text{m/s}, -3.7\text{ deg}, 0\text{ deg/s}, 235\text{RPM})$.

Case 4. $\mathbf{s_0} = (-80\text{m}, 10\text{m/s}, 0\text{m/s}, -2.4\text{ deg}, 0\text{ deg/s}, 235\text{RPM})$.

Trajectories are obtained following procedures described in Section 2. We apply a Hermite interpolation of 20 points sampled from the preliminary trajectories. Note that such number of points can be regarded as sufficient, considering there are 10 points in [11] and 16 points in [12]. Results are shown in Figure 4.

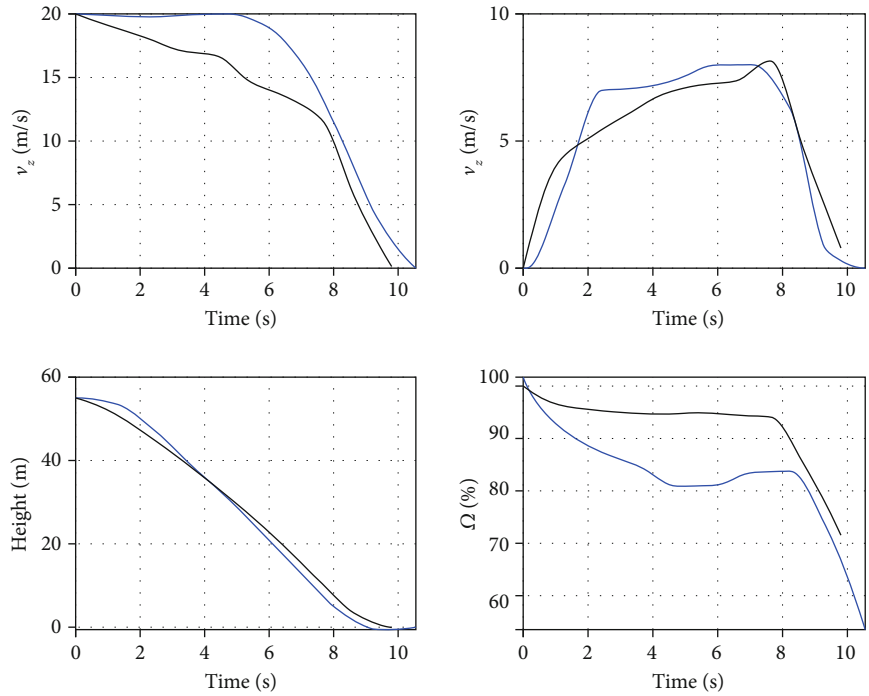

Figure 5: Comparisons of trajectories with results from SQP methods. Black lines are trajectories using FT-based DP method, and blue lines are trajectories using SQP method.

Table 2: Time cost for each trajectory generation.

Case number	1	2	3	4	Average
Algorithm 1 time cost, s	0.62	0.59	0.61	0.59	0.60
Algorithm 2 time cost, s	9.12	11.54	11.21	13.74	11.40

As Figure 4 shows, the helicopter enters a steady descent quickly, and begins to decrease at about 20 meters above the ground. Such behaviours agree generally with [9]. For high initial speed cases, the forward speed is decreased immediately after the engine fails, in order to get the helicopter prepared for the final landing. Such behaviours are in accordance with [5]. All the trajectories terminate within the landing constraints defined in (28). Thus, the trajectory planning method using FT-based DP algorithms is able to generate successful autorotation trajectories that satisfy the helicopter autorotation dynamics described in Section 2. Furthermore, comparisons are made with results obtained from Sequential Quadratic Programming (SQP) using the SNOPT software package [34]. In specific, besides the landing specifications assumed in (28), path constraints are required, mainly as $0\text{m/s} \le v_z \le 9\text{m/s}, -30\ \text{deg} \le \theta \le 30\ \text{deg}, \bar{\Omega} \ge 70\%$. A typical result is demonstrated using initial conditions of Case 1 and is shown in Figure 5.

Both trajectories in Figure 5 lead to safe autorotation landing, considering landing constraints are all satisfied. Besides, the two trajectories show similarities in terms of the rate of descent. The main difference is that the trajectory using the FT-based DP method tends to maintain the rotating speed by entering the steady autorotation right after the engine failure, while the trajectory obtained by SQP method appears to be more agile since the rotor speed does not reach the lower limit of the path constraint. In this respect, the autorotation trajectory using the FT-based DP method is more conservative in terms of keeping a higher rotor speed.

Both the above trajectory planning results are obtained using one core of a 3.20 GHz Inter i7-8700 CPU. Time costs for each trajectory generation using the FT-based DP method (shown as Algorithm 1) and SQP method (shown as Algorithm 2) are listed in Table 2. The average time cost of the FT-based DP method is 0.60 s, which indicates that the method is real-time feasible.

Before entering autorotation, the time delay of disengaging the clutch is a key factor that determines whether the subsequent maneuvers lead to a safe landing. A long time delay makes autorotation a tough task because of a low rotor speed. Here, the influence of time delay is considered by setting lowered rotating speeds. Initial conditions are given by $\mathbf{s}_0 = (-90\text{m}, 20\text{m/s}, 0\text{m/s}, -3.7\ \text{deg}, 0\ \text{deg/s}, \Omega)$, where $\Omega = \{100\%, 95\%, 90\%\} \cdot 235\text{RPM}$ for each trajectory.

From Figure 6, we can see that for lowered rotor speeds caused by different time delays, the trajectory planner is still capable of generating trajectories without modifying any cost function parameter. The rotating speed is recovered along the trajectory by certain maneuvers.

As for real-time applications, because the trajectory planner is able to generate a global trajectory by any specified initial condition, the computing speed shown in Table 2 can be considered real-time applicable with such hardware. The trajectory computing time cost in our study is about 0.6 seconds,

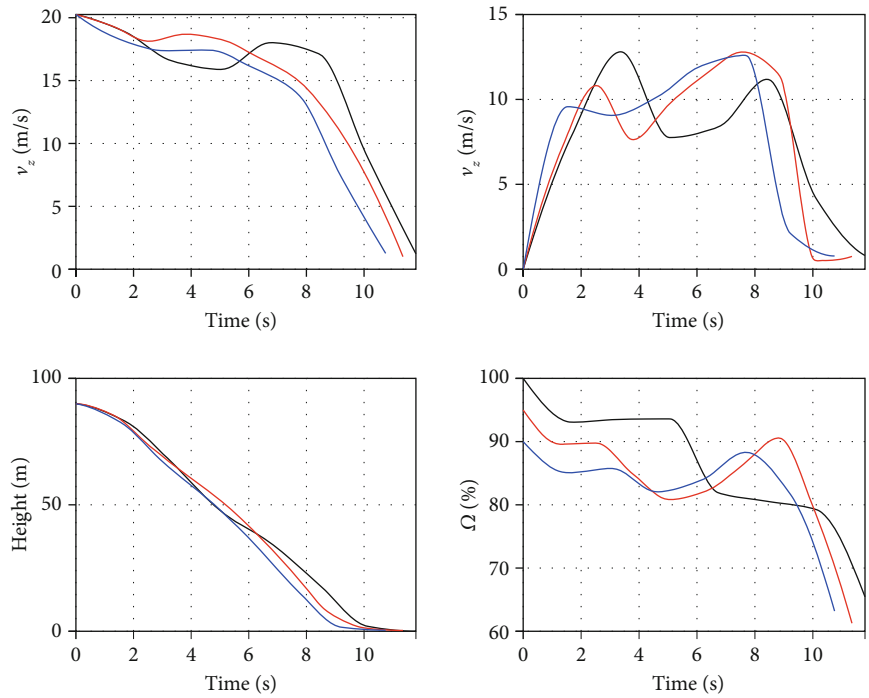

Figure 6: Entering autorotation with different rotor speed. Black lines for 100%Ω, red lines for 95%Ω, and blue lines for 90%Ω.

which is similar to the time of a fast human pilot reaction. However, as mentioned above, the time delay is a key factor of the autorotation procedure, which should be shortened with best efforts. Therefore, although the trajectory planner is shown to be able to deal with situations of different time delays, an update frequency of at least 1.5 Hz is recommended for real-time applications, considering the time cost of other parts (such as detection of engine failure). Thus, we recommend running such algorithms with CPU of at least 3.1 GHz or higher.

5.2. Trajectory-Tracking Simulation with the TRUST 6-DOF Helicopter Dynamic Model. In this part, for validation of the dynamic feasibility of the trajectory, we demonstrate a six-degree-of-freedom flight simulation, with the trajectory planner described in Section 2 and the trajectory-tracking controller described in Section 3. For brevity, control parameters can be found in the Appendix. As mentioned before, the simulation is based on a six-degree-of-freedom nonlinear multibody-based helicopter model of S-58, with a VRS correction of the Pitt-Peters dynamic inflow and an accurate dynamic description of the blade's flap motions.

We demonstrate the simulation results of initial conditions from Case 1. The initial conditions of states $\mathbf{s}$ are given by

$$\mathbf{s_0} = (-55m, 20m/s, 0m/s, -0.7 \text{ deg}, 0 \text{ deg}/s, 235\text{RPM}). \quad (30)$$

We assume the engine failure starts at $t = 0$s in the forward flight, and the simulations are terminated once the height descends to zero.

Simulation results are shown in Figure 7. Remember that v_{zb} is positive downward in the body axes and θ is positive in the nose-up direction. The red lines denote the reference values that are needed to be tracked. The black lines denote the 6-DOF nonlinear helicopter model responses. The blue line indicates that the helicopter is close to the ground, and the controller is reset to make a slow landing. The value of the reset height is chosen as 1.0 m by simulation experiments.

As the results shown in Figure 7, the velocities and attitude angles of the helicopter are $v_{xb} = -0.14$ m/s, $v_{yb} = -0.42$m/s, $v_z = 1.15$m/s, $\phi = 3.21$ deg, and $\theta = -2.33$ deg when landing, which satisfy the safe autorotation landing specifications defined above. The height history and rotating speed of the main rotor are shown in lower right and the upper right panel of Figure 7, which indicate that the rotating speed is still beyond 75%, although the ground effect is neglected in the six-degree-of-freedom nonlinear helicopter model. Notice that the performance of horizontal speed is not as good as the vertical channel. However, we do not set horizontal distance as any kind of target variable in this study. We also find that increasing horizontal controlling gains affects the vertical channel instead. Apparently, a more sophisticated model-based controller should improve the tracking performance, but our purpose of validating the trajectories is achieved. Simulation results show that following the trajectories generated by the trajectory planner, the helicopter is successfully guided to a safe landing. Thus the trajectory is both real-time feasible and dynamic feasible.

6. Conclusions

This paper demonstrates a trajectory planning method for autorotation, which is real-time feasible and guarantees a

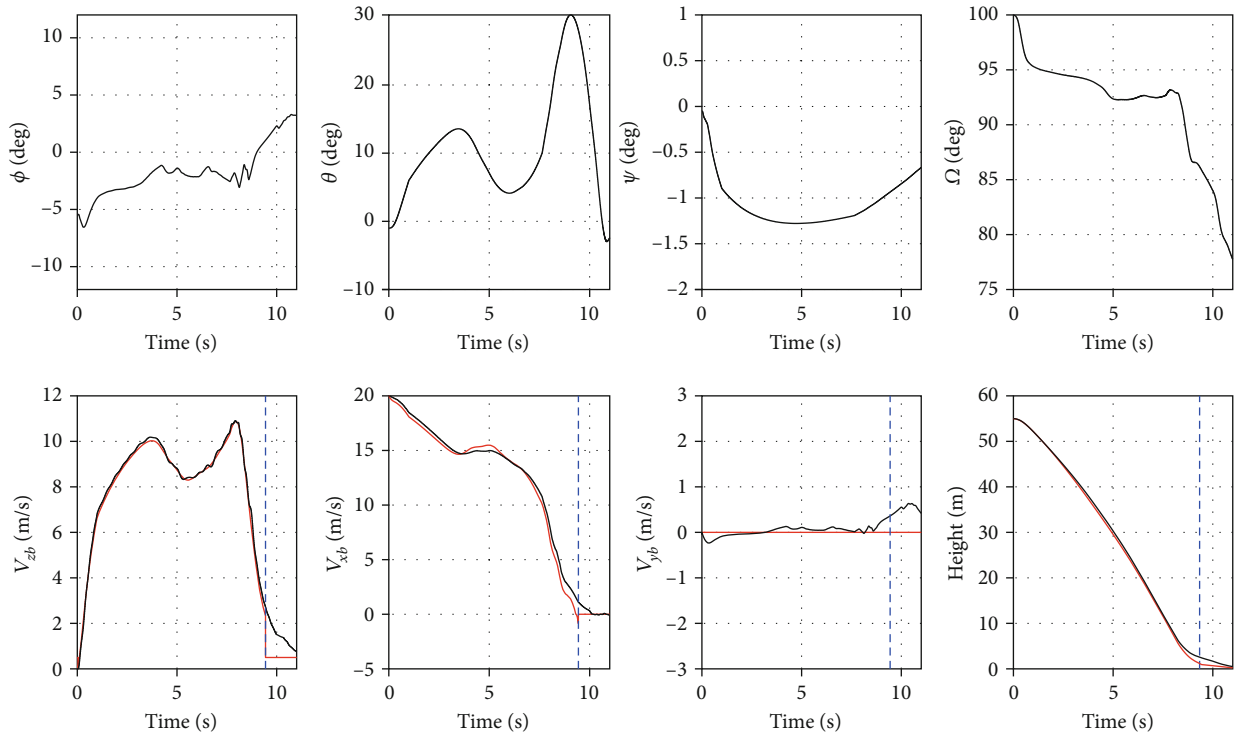

Figure 7: Height, rotor speed, attitude of Euler angles, and velocities in the body axes during simulation. Black lines denote simulation results, red lines denote reference values of the trajectory-tracking controller, and blue lines denote controller reset for landing.

Table 3: Parameters of the NLSEF.

Controller channel	$\lvert b_0 \rvert$	β_1	β_2	β_3	α_1	α_2	α_3
v_{xb}	113	0.34	0.13	0.05	1.3	1.76	2.4
v_{yb}	158	0.20	0.08	0.02	0.8	1.1	1.5
v_{zb}	1.6	0.12	0.04	0.01	0.2	0.58	0.8
ψ	1	0.15	—	—	0.8	—	—

Table 4: Parameters of the ESO and TD.

Controller channel	r	β_{01}	β_{02}	β_{03}	β_{04}	δ/h_0
v_{xb}	100	100	300	1000	1800	6
v_{yb}	100	100	300	1000	1800	6
v_{zb}	100	100	300	1000	1800	6
ψ	0.5	—	—	—	—	—
$v_{\mathrm{xb\_ref}}$	1	—	—	—	—	—
$v_{\mathrm{yb\_ref}}$	1	—	—	—	—	—
$v_{\mathrm{zb\_ref}}$	0.8	—	—	—	—	—

Table 5: Parameters of the inner loop.

Controller channel	K	I	Damping
q	0.73	0.07	1.12
p	0.23	0.019	0.32
r	3.0	0.36	4.8

strict satisfaction of specified helicopter dynamics along the trajectory. A trajectory-tracking controller is demonstrated to ensure the helicopter fly along the trajectories. Successful autorotation trajectories with various initial conditions are shown and discussed, and the time cost of trajectory generation procedures is around 0.60 s. A comprehensive validation is made based on a six-degree-of-freedom high-fidelity nonlinear S-58 model using the TRUST simulator. Simulation results show that although the horizontal velocities are not tracked as well as the vertical channel, the trajectories generated by the trajectory planner and the trajectory tracker are capable of guiding a helicopter into a successful autorotation. Thus, the trajectory planner is real-time feasible, and the generated trajectories are dynamic feasible. Further potential improvements for the trajectory planning method could be selecting the derivatives of the control as control inputs of the three-degree-of-freedom helicopter dynamics, extending the two-dimensional helicopter dynamics to a three-dimensional helicopter dynamic model, and automatic tuning of the weight functions.

Appendix

A. Trajectory-Tracking Controller Parameters

The controller parameters described in Section 3 are listed below. Parameters of the NLSEF are shown in Table 3.

Parameters of the ESO and TD are listed in Table 4.

Parameters of the inner loop are listed in Table 5.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work has been supported by the Zhuhai Longhua Helicopter Science and Technology Co., Ltd.

References

[1] B. L. Aponso, E. N. Bachelder, and D. Lee, "Automated autorotation for unmanned rotorcraft recovery," in *AHS International Specialists' Meeting - Unmanned Rotorcraft: Design, Control and Testing, Proceedings*, pp. 21–33, 2005.

[2] S. Taamallah, "A qualitative introduction to the vortex-ring-state, autorotation, and optimal autorotation," in *36th European Rotorcraft Forum, ERF 2010*, pp. 464–492, Paris, France, 2010.

[3] P. Bibik and J. Narkiewicz, "Helicopter optimal control after power failure using comprehensive dynamic model," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, pp. 1354–1362, 2012.

[4] M. W. Floros, "Descent analysis for rotorcraft survivability with power loss," in *American Helicopter Society 65th Annual Forum*, Grapevine, TX, USA, May 2009.

[5] A. Y. Lee, A. E. Bryson Jr., and W. S. Hindson, "Optimal landing of a helicopter in autorotation," *Journal of Guidance, Control, and Dynamics*, vol. 11, no. 1, pp. 7–12, 1988.

[6] E. N. Bachelder and B. L. Aponso, "An autorotation flight director for helicopter training," in *Annual Forum Proceedings-American Helicopter Society*, vol. 59, no. 2, 2003American Helicopter Society, Inc, 2003.

[7] Y. Okuno and K. Kawachi, "Optimal control of helicopters following power failure," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 1, pp. 181–186, 1994.

[8] C. L. Bottasso, A. Croce, D. Leonello, and L. Riviello, "Rotorcraft trajectory optimization with realizability considerations," *Journal of Aerospace Engineering*, vol. 18, no. 3, pp. 146–155, 2005.

[9] W. Johnson, "Helicopter optimal descent and landing after power loss," NASA TM 73244, Ames Research Center, National Aeronautics and Space Administration, Moffett Field, CA, USA, 1977.

[10] E. B. Carlson, "An analytical methodology for category a performance prediction and extrapolation," in *Annual Forum Proceedings-American Helicopter Society*, vol. 57, no. 2, 2001American Helicopter Society, Inc, 2001.

[11] A. A. Jhemi, E. B. Carlson, Y. J. Zhao, and R. T. N. Chen, "Optimization of rotorcraft flight following engine failure," *Journal of the American Helicopter Society*, vol. 49, no. 2, pp. 117–126, 2004.

[12] S. Taamallah, X. Bombois, and P. M. J. Van den Hof, "Trajectory planning and trajectory tracking for a small-scale helicopter in autorotation," *Control Engineering Practice*, vol. 58, pp. 88–106, 2017.

[13] W. Johnson, *Helicopter Theory*, Courier Corporation, 2012.

[14] A. A. Gorodetsky, S. Karaman, and Y. M. Marzouk, "Function-train: a continuous analogue of the tensor-train decomposition," 2015, http://arxiv.org/abs/1510.09088.

[15] A. Gorodetsky, S. Karaman, and Y. Marzouk, "High-dimensional stochastic optimal control using continuous tensor decompositions," *The International Journal of Robotics Research*, vol. 37, no. 2-3, pp. 340–377, 2018.

[16] A. A. Gorodetsky, *Continuous Low-Rank Tensor Decompositions, with Applications to Stochastic Optimal Control and Data Assimilation*, Diss. Massachusetts Institute of Technology, 2017.

[17] H. Kushner and P. G. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time*, vol. 24, Springer Science & Business Media, 2013.

[18] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[19] I. Oseledets and E. Tyrtyshnikov, "TT-cross approximation for multidimensional arrays," *Linear Algebra and its Applications*, vol. 432, no. 1, pp. 70–88, 2010.

[20] Y. Okuno, K. Kawachi, A. Azuma, and S. Saito, "Analytical prediction of height-velocity diagram of a helicopter using optimal control theory," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 2, pp. 453–459, 1991.

[21] U. T. P. Arnold, J. D. Keller, H. C. Curtiss, and G. Reichert, "The effect of inflow models on the predicted response of helicopters," *Journal of the American Helicopter Society*, vol. 43, no. 1, pp. 25–36, 1998.

[22] C. H. E. E.-S. Chow and J. N. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control," *IEEE Transactions on Automatic Control*, vol. 36, no. 8, pp. 898–914, 1991.

[23] J. Han, "From PID to active disturbance rejection control," *IEEE transactions on Industrial Electronics*, vol. 56, no. 3, pp. 900–906, 2009.

[24] C. Jiang, H. W. Wang, J. K. Li, and L. J. Li, "Trajectory-tracking hybrid controller based on ADRC and adaptive control for unmanned helicopters," *Chinese Journal of Engineering*, vol. 11, 2017.

[25] C. Jiang, *A Model Reference-Active Disturbance Rejection Hybrid Trajectory Tracking Controller for Unmanned Helicopter*, Diss. Tsinghua University, 2017.

[26] S. Deng, C. Jiang, Y. Wang, and H. Wang, "Helicopter flight dynamics simulation with continues-time unsteady vortex lattice-free wake and multibody dynamics," in *The Proceedings of the 2018 Asia-Pacific International Symposium on Aerospace Technology (APISAT 2018). APISAT 2018. Lecture Notes in Electrical Engineering, vol 459*, X. Zhang, Ed., Springer, Singapore, 2018.

[27] D. A. Peters and N. HaQuang, "Dynamic inflow for practical applications," *Journal of the American Helicopter Society*, vol. 33, no. 4, 1998.

[28] D. A. Peters and C. He, "Technical note: modification of mass-flow parameter to allow smooth transition between helicopter and windmill states," *Journal of the American Helicopter Society*, vol. 51, no. 3, pp. 275–278, 2006.

[29] J. R. Majhi and R. Ganguli, "Modeling helicopter rotor blade flapping motion considering nonlinear aerodynamics," *Computer Modeling in Engineering and Sciences*, vol. 27, no. 1/2, pp. 25-26, 2008.

[30] G. H. Gaonkar and D. A. Peters, "Effectiveness of current dynamic-inflow models in hover and forward flight," *Journal of the American Helicopter Society*, vol. 31, no. 2, pp. 47–57, 1986.

[31] G. D. Padfield, *Helicopter Flight Dynamics*, John Wiley & Sons, Chichester, UK, 2008.

[32] W. J. Hanley, G. DeVore, and S. Martin, *An Evaluation of the Height Velocity Diagram of a Heavyweight, High Rotor Inertia, Single Engine Helicopter. No. FAA-ADS-84*, Federal Aviation Administration Washington Dc Aircraft Development Service, 1966.

[33] J. Scheiman, *A tabulation of helicopter rotor-blade differential pressures, stresses, and motions as measured in flight. Vol. 952*, National Aeronautics and Space Administration, 1964.

[34] P. E. Gill, W. Murray, and M. A. Saunders, *User's Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming*, Report NA 05-2, Department of Mathematics, University of California, San Diego, CA, USA, 2006.