Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 464056, 8 pages
http://dx.doi.org/10.1155/2014/464056

Research Article

Trajectory Evaluation of Rotor-Flying Robots Using Accurate Inverse Computation Based on Algorithm Differentiation

Yuqing He, Yingjun Zhou, and Jianda Han

State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China

Correspondence should be addressed to Yuqing He; heyuqing@sia.cn

Received 26 March 2014; Revised 9 June 2014; Accepted 11 June 2014; Published 13 August 2014

Academic Editor: Fatih Yaman

Autonomous maneuvering flight control of rotor-flying robots (RFR) is a challenging problem due to the highly complicated structure of its model and significant uncertainties regarding many aspects of the field. As a consequence, it is difficult in many cases to decide whether or not a flight maneuver trajectory is feasible. It is necessary to conduct an analysis of the flight maneuvering ability of an RFR prior to test flight. Our aim in this paper is to use a numerical method called algorithm differentiation (AD) to solve this problem. The basic idea is to compute the internal state (i.e., attitude angles and angular rates) and input profiles based on predetermined maneuvering trajectory information denoted by the outputs (i.e., positions and yaw angle) and their higher-order derivatives. For this purpose, we first present a model of the RFR system and show that it is flat. We then cast the procedure for obtaining the required state/input based on the desired outputs as a static optimization problem, which is solved using AD and a derivative based optimization algorithm. Finally, we test our proposed method using a flight maneuver trajectory to verify its performance.

1. Introduction

The autonomous rotor-flying robot (RFR) is one of the frontier research topics in the field of robotics. Extensive research has been conducted on issues related to RFR, including flight control [1, 2], path/trajectory planning [3, 4], and intelligent navigation [5, 6].

Flight maneuvering is a very important ability for an RFR system because it can greatly extend the system's field of applications. However, the flight maneuver control of an RFR is a challenging and absorbing problem in the field of autonomous RFR research [2]. This can be explained from the following two points. On the one hand, the model of an RFR system is an extremely complicated structure [7, 8], designing a suitable controller for which is very difficult. On the other hand, flight maneuvering requires that both the interval states (i.e., the attitude angles and the angular rates) and the control surfaces to have a large scope variation (or even approach the largest permitted value) [9]. This may produce a great deal of uncertainty and thus presents that this flight maneuvering is dangerous and difficult to be implemented and thus requires a much more robust controller than usual [10].

Most work on flight maneuver control, such as [11, 12], is based on successful flight demonstrations and grounded in learning techniques used to update the control laws. These methods have successfully pushed the envelope of what is possible with autonomous control, but they lack performance guarantees and are limited to situations where safety is not a critical concern. Thus, it is very important to assess the flight maneuvering ability of an RFR system prior to designing a controller and conducting tests. Some research has been conducted in this area. For instance, research in [2] is based on reachability analysis and explores aerobatic maneuvers and multiple vehicle collision avoidance of the RFR systems. This involves designing a sequence of modes and conducting safe transition from one to the next. However, reachability analysis is very time consuming, due to which the proposals in [2] are viable only for simplified models. Similarly, [9] involves a feasibility analysis of planned trajectories by using a greatly simplified dynamics model (without considering angular dynamics). Both the positions and attitude angles are predetermined and then used to evaluate the feasibility of the planned trajectories. Although the algorithm is simple and easily implemented, it is unfit for many planning algorithms

since for which it is difficult to obtain the positions and attitude angles simultaneously.

In this paper, based on a full state dynamics model, we investigate the feasibility analysis problem of a flight maneuvering trajectory. This is actually an inverse computation problem, that is, to compute both the interval states and the inputs based on predetermined position, yaw angle, and their higher-order derivatives. In our work, we model it as a static optimization problem and then use a derivative-based optimization algorithm to obtain an accurate solution of it. We use algorithm differentiation (AD) to obtain precise higher-order derivatives of the nonlinear cost function. The advantages of our proposed algorithm are as follows: (1) only with desired positions and their derivatives, which is the case for many planning algorithm, the feasibility of the trajectories (including inner states and inputs) can be evaluated; (2) inverse computation can be very accurate because AD is a numerical algorithm that can precisely obtain the derivative of a nonlinear function. Thus, an accurate pointwise numerical feasibility analysis of planned trajectories becomes possible.

The remainder of this paper is organized as follows. We first present a model of the RFR system and show that it is flat by taking the three positions and the yaw angle as a group of flat outputs. This allows us to model the trajectory evaluation problem as an optimization problem by taking into account the dynamical model, which is solved by using the AD algorithm along with a derivative-based optimization algorithm. Finally, in order to test our proposed method, we use it to analyze a flight maneuvering trajectory.

2. Dynamical Model of the RFR System

Usually, the complete dynamical model of an RFR system can be divided into three parts: the actuator dynamics, the aerodynamics, and the rigid body dynamics. In this paper, only the rigid body dynamics as shown in the following equation (1) is considered [3]:

$$\begin{bmatrix} \dot{p} \\ \dot{v}^p \\ \dot{\Theta} \\ \dot{\omega}^b \end{bmatrix} = \begin{bmatrix} v^p \\ \frac{1}{m} R f^b \\ \Psi \omega^b \\ J^{-1}\left(\tau^b - \omega^b \times J\omega^b\right) \end{bmatrix}, \tag{1}$$

where $p = [x\ y\ z]^T \in R^3$ and $v^p = [\dot{x}\ \dot{y}\ \dot{z}] \in R^3$ are translational position and velocity vector of the RFR in the inertia frame; R is the rotation matrix between the body frame and the inertia frame; ω^b is angular velocity vector; $\Theta = [\phi\ \theta\ \psi]^T$ is Euler angle vector; m and J are, respectively, the mass and inertia matrix of the RFR; Ψ is the transformation matrix from angular velocity vector in body frame to angular velocity vector in inertia frame; and f^b and τ^b are force and moment of the RFR presented in body frame. Different kinds of RFR have very different aerodynamics, that is, the relationship between the force/moment and the control surfaces. Here we focus on the quadracopter flying robot, whose driving forces and moments can be denoted as follows:

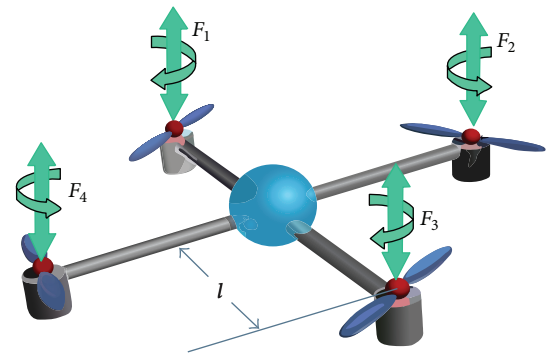

Figure 1: Sketch of quadrocopter.

$$f^b = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ T_M \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix},$$
$$\tau^b = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} l(F_4 - F_2) \\ l(F_3 - F_1) \\ l(F_1 - F_2 + F_3 - F_4) \end{bmatrix}, \tag{2}$$

where F_1, F_2, F_3, and F_4 are force produced by four rotors, respectively, and l is the distance between each rotor center and the RFR's center of gravity (see Figure 1).

Thus, define inputs as

$$u = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \triangleq \begin{bmatrix} \ddot{T}_M \\ M_x \\ M_y \\ M_z \end{bmatrix}. \tag{3}$$

System (1) can be rewritten as

$$\begin{aligned} \ddot{x} &= \frac{\cos\psi \sin\theta \cos\varphi + \sin\psi \sin\varphi}{m} T_M, \\ \ddot{y} &= \frac{\sin\psi \sin\theta \cos\varphi - \sin\varphi \cos\psi}{m} T_M, \\ \ddot{z} &= -g + \frac{\cos\theta \cos\varphi}{m} T_M, \\ \ddot{\varphi} &= \frac{I_y - I_z}{I_x} \dot{\theta}\dot{\psi} + \frac{1}{I_x} u_2, \\ \ddot{\theta} &= \frac{I_z - I_x}{I_y} \dot{\varphi}\dot{\psi} + \frac{1}{I_y} u_3, \\ \ddot{\psi} &= \frac{I_x - I_y}{I_z} \dot{\theta}\dot{\varphi} + \frac{1}{I_z} u_4, \\ \ddot{T}_M &= u_1. \end{aligned} \tag{4}$$

It can be easily shown that system (4) is input-to-state feedback linearizable through defining the following state vector:

$$\left[x \;\; y \;\; z \;\; \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \ddot{x} \;\; \ddot{y} \;\; \ddot{z} \;\; \dddot{x} \;\; \dddot{y} \;\; \dddot{z} \;\; \psi \;\; \dot{\psi}\right]^T. \tag{5}$$

From (1) and (2), we have

$$\begin{aligned}
\mathbf{p}^{(4)} &= -\frac{1}{m}R\left[\boldsymbol{\omega}^b \times \left(\boldsymbol{\omega}^b \times \begin{bmatrix} 0 \\ 0 \\ T_M \end{bmatrix}\right)\right] + \frac{1}{m}R\begin{bmatrix} 0 & -T_M & 0 \\ T_M & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\
&\quad \times J^{-1}\left(\tau^b - \boldsymbol{\omega}^b \times J\boldsymbol{\omega}^b\right) \\
&\quad - \frac{2}{m}R\left\{\boldsymbol{\omega}^b \times \begin{bmatrix} 0 & 0 & \dot{T}_M \end{bmatrix}^T\right\} - \frac{1}{m}R[0 \;\; 0 \;\; 1]^T\ddot{T}_M, \\
\ddot{\psi} &= -\begin{bmatrix} 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{bmatrix} J^{-1}\left(\boldsymbol{\omega}^b \times J\boldsymbol{\omega}^b - \tau^b\right) \\
&\quad + \frac{\left(q\dot{\phi}\cos\phi - r\dot{\phi}\sin\phi\right)}{\cos\theta} \\
&\quad \times \left[p + 2q\sin\phi\tan\theta + 2r\cos\phi\tan\theta\right]
\end{aligned} \tag{6}$$

and the system can be transformed into the following linear form:

$$\begin{aligned}
p^{(4)} &= \begin{bmatrix} x^{(4)} \\ y^{(4)} \\ z^{(4)} \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}, \\
\ddot{\psi} &= v_4
\end{aligned} \tag{7}$$

with the feedback linearization controller

$$u = \begin{bmatrix} g_2 \\ g_3 \end{bmatrix}^{-1}\left(\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} - \begin{bmatrix} f_2 \\ f_3 \end{bmatrix}\right), \tag{8}$$

where

$$\begin{aligned}
f_2 &= -\frac{2}{m}\dot{T}_M R\left(\omega^b \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right) \\
&\quad - \frac{1}{m}T_M R\left[\omega^b \times \left(\omega^b \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right)\right] \\
&\quad - \frac{1}{m}T_M R\left\{\left[J^{-1}\left(\omega^b \times J\omega^b\right)\right] \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right\}; \\
g_2 &= \left[-\frac{1}{m}R[0 \;\; 0 \;\; 1]^T \;\; -\frac{T_M}{m}R\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} J^{-1}\right]; \\
f_3 &= -\begin{bmatrix} 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{bmatrix} J^{-1}\left(\omega^b \times J\omega^b\right) \\
&\quad + \left(q\cos\phi - r\sin\phi\right) \\
&\quad \times \sec\theta\left[p + 2q\sin\phi\tan\theta + 2r\cos\phi\tan\theta\right]; \\
g_3 &= \left[0 \;\; \begin{bmatrix} 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{bmatrix} J^{-1}\right].
\end{aligned} \tag{9}$$

That means system (1) is flat and that the $[p^T, \psi]$ are the corresponding flat outputs.

3. Automatic Differential Algorithm Based Inverse Computation

AD is a numerical method that is effective to accurately compute the derivatives of some complicated nonlinear function. In this section, we will briefly introduce how to conduct the inverse computation of a nonlinear flat system using AD algorithm [13].

Consider the following nonlinear system:

$$\begin{aligned}
\dot{x} &= f(x), \\
y &= h(x).
\end{aligned} \tag{10}$$

It has been shown that the inverse computation problem of system (10), that is, to obtain a state vector x_d with $h(x_d) = y_d$ (a predefined output), is solvable if $y = h(x)$ is a flat output of system (1) [14].

Derivatives of y with respect to x are necessary to effectively solve the preceding inverse problem. In the following contents of this section, the basic idea on how to obtain $\partial y/\partial x$ using AD algorithm will be introduced.

For this purpose, we firstly use the following Taylor series to denote the state vector $x(t)$:

$$x(t) = x_0 + x_1 t + x_2 t^2 + \cdots + x_d t^d. \tag{11}$$

Provided that $f(*)$ is d times continuously differentiable, $z = f(x)$ can also be expressed by Taylor series as follows:

$$z(t) = z_0 + z_1 t + z_2 t^2 + \cdots + z_d t^d + \mathrm{O}\left(t^{d+1}\right). \tag{12}$$

It has been shown that each Taylor coefficient z_i ($i = 0, 1, \ldots, d$) is uniquely determined by the coefficients x_i ($i = 0, 1, \ldots, d$) as follows [15, 16]:

$$\begin{aligned}
z_0 &= f(x_0) \\
z_1 &= \frac{\partial f(x_0)}{\partial x_0}x_1 \\
z_2 &= \frac{\partial f(x_0)}{\partial x_0}x_2 + \frac{1}{2}\frac{\partial^2 f(x_0)}{\partial x_0 \partial x_0}x_1 x_1 \\
&\vdots
\end{aligned} \tag{13}$$

Thus, if x_i $(i = 0, 1, \ldots, d)$ in (11) are preknown, all z_i $(i = 0, 1, \ldots, d)$ can be obtained based on some derivatives, which can be computed by basic "forward mode" of AD [13]. Similarly, the following A_{j-i} can also be derived using "reverse mode" of AD [13–15]:

$$A_{j-i}\left(j \geq i\right) \triangleq \frac{\partial z_{j-i}}{\partial x_0}. \tag{14}$$

Furthermore, based on (10), we have

$$\begin{aligned} z\left(t\right) &= z_0 + z_1 t + z_2 t^2 + \cdots + z_d t^d + \mathrm{O}\left(t^{d+1}\right) \\ &= f\left(x\right) = \dot{x} = x_1 + 2x_2 t + \cdots + d x_d t^{d-1}; \end{aligned} \tag{15}$$

that is,

$$z_i \equiv \left(i+1\right) x_{i+1}, \quad i = 0, \ldots, d-1. \tag{16}$$

Through (16) and (13), we can easily compute all z_i based on x_0.

Define a new matrix as

$$B_k \triangleq \frac{dx_{k+1}}{dx_0} = \frac{1}{k+1}\frac{dz_k}{dx_0} = \frac{1}{k+1}\sum_{j=0}^{k}\frac{\partial z_k}{\partial x_j}\frac{dx_j}{dx_0}. \tag{17}$$

From [15], we have

$$\frac{\partial z_k}{\partial x_j} = \frac{\partial z_{k-j}}{\partial x_0}. \tag{18}$$

Thus, B_k can be iteratively obtained as follows:

$$B_k = \frac{1}{k+1}\sum_{j=0}^{k}\frac{\partial z_{k-j}}{\partial x_0}\frac{dx_j}{dx_0} = \frac{1}{k+1}\left(A_k + \sum_{j=0}^{k} A_{k-j}B_{j-1}\right). \tag{19}$$

Furthermore, if we denote $y(t)$ as

$$y\left(t\right) = y_0 + y_1 t + y_2 t^2 + \cdots + y_d t^d + \mathrm{O}\left(t^{d+1}\right), \tag{20}$$

then the same process can be used to compute y through the second equation of (10); that is, with a preknown x_0, compute y_i by forward mode and compute the following derivatives C_{j-i} by reverse mode:

$$C_{j-i} \triangleq \frac{\partial y_j}{\partial x_i}. \tag{21}$$

Also, it is not difficult to show that the derivatives of output with respect to the states can be denoted as follows:

$$D_k \triangleq \frac{dy_k}{dx_0} = \sum_{j=0}^{k}\frac{\partial y_k}{\partial x_j}\frac{dx_j}{dx_0} = C_k + \sum_{j=1}^{k} C_{k-j}B_{j-1}. \tag{22}$$

Up to now, we have shown that, with a predefined state x_0, both the corresponding outputs $y(x_0)$ and its higher-order derivatives with respect to time, as well as the derivatives with respect to state x_0 (22), can be obtained through AD algorithm. These results can be further used to conduct the inverse computation of system (1), and the detailed steps are given as follows.

AD Based Inverse Computation Algorithm

Initialization. d (the highest order of the outputs' derivatives); γ (step length of searching algorithm); $k = 0$ (iterative number).

Step 1. Compute the output vector $Y(x_k)$ using forward mode of AD algorithm at each time instant as a nonlinear map; that is,

$$\Theta : x^k \longrightarrow Y\left(x^k\right) \triangleq \begin{pmatrix} y_0\left(x^k\right) \\ \vdots \\ y_d\left(x^k\right) \end{pmatrix}. \tag{23}$$

Step 2. Based on (22), AD can be used to obtain the partial derivative of Y with respect to x_0; that is,

$$\Theta'\left(x^k\right) = \frac{\partial Y}{\partial x^k} = \begin{pmatrix} D_0 \\ \vdots \\ D_d \end{pmatrix}. \tag{24}$$

Step 3. Update the state x using the following searching iteration:

$$\begin{aligned} x^{k+1} &\triangleq x^k - \gamma\left(\Xi'\left(x^k\right)\right)^{+}\Xi\left(x^k\right) \\ &= x^k - \gamma\left(\Xi'\left(x^k\right)\right)^{+}\left[\Theta\left(x^k\right) - Y_d\right] \\ &= x^k - \gamma\left(\Theta'\left(x_0\right)\right)^{+}\left[\Theta\left(x^k\right) - Y_d\right], \end{aligned} \tag{25}$$

where superscript "+" means the pseudoinverse of a matrix; that is,

$$\begin{aligned} Q^{+} &= \left(Q^T Q\right)^{-1} Q^T, \\ \Xi\left(x\right) &= \Theta\left(x\right) - Y_d. \end{aligned} \tag{26}$$

Step 4. Go to Step 1 until the terminal conditions are satisfied.

If the system is nonautonomous, that is, it can be rewritten as following form,

$$\begin{aligned} \dot{x} &= f\left(x, u\right), \\ y &= h\left(x\right). \end{aligned} \tag{27}$$

We can first extend it into the following form:

$$\begin{aligned} \dot{x} &= f\left(x, u\right), \\ \dot{u} &= 0, \\ y &= h\left(x\right) \end{aligned} \tag{28}$$

and the new system can be rewritten as

$$\begin{aligned} \dot{\overline{x}} &= F\left(\overline{x}\right), \\ y &= H\left(\overline{x}\right), \end{aligned} \tag{29}$$

where

$$\overline{x} = \begin{bmatrix} x \\ u \end{bmatrix}. \tag{30}$$

Thus, the inverse computation can be conducted using the preceding algorithm.

As for the proposed quadrocopter system introduced in Section 2, this algorithm can be directly utilized through defining the following output vector:

$$Y \triangleq \left[p^T \ \dot{p}^T \ \ddot{p}^T \ \dddot{p}^T \ \left(p^{(t)}\right)^T \ \psi \ \dot{\psi} \ \ddot{\psi} \right]^T_{18\times 1}. \tag{31}$$

4. Evaluation of Circle Flying Maneuver

In this section, a so-called "Circle Maneuver," that is, the RFR system flies along a circle as shown in the following equation (32) and Figure 2 [10], of the Quadrocopter is taken as an example to show the validity of the proposed algorithm. Consider

$$\begin{aligned} \psi_d &= \pi + \frac{t}{T} \times 2\pi, \\ x_d &= 50 \sin\left(\frac{t}{T} \times 2\pi\right), \\ y_d &= 50 \cos\left(\frac{t}{T} \times 2\pi\right), \\ z_d &= 0, \end{aligned} \tag{32}$$

where T is the period of the circle maneuver.

The parameters of the RFR system are as follows:

$$m = 9.5\,\text{kg}$$
$$J = \begin{bmatrix} 0.1364 & 0 & 0 \\ 0 & 0.5782 & 0 \\ 0 & 0 & 0.6306 \end{bmatrix} \text{N} \cdot \text{m}. \tag{33}$$

The trajectory feasibility is decided by some constrains on both inputs and states. Here the following constrains are considered.

Input Constrains. Consider

$$\begin{aligned} T_M &\in [49.4, 190.8]\,\text{N}, \qquad M_x \in [-25.5, 25.5]\,\text{N} \cdot \text{m}, \\ M_y &\in [-40.2, 40.2]\,\text{N} \cdot \text{m}, \qquad M_z \in [-34.6, 34.6]\,\text{N} \cdot \text{m}. \end{aligned} \tag{34}$$

Angular Velocity Constrains. Consider

$$p, q \in [-0.4, 0.4]\,\text{rad/s}, \qquad r \in [-1.45, 1.45]\,\text{rad/s}. \tag{35}$$

Attitude Constrains. Consider

$$\theta, \phi \in [-0.5, 0.5]\,\text{rad}. \tag{36}$$

In this paper, the software toolbox LIEDRIVERS [17] based on ADOL-C is used as the basic AD algorithm, and the optimization searching is implemented using the golden sector algorithm, instead of the direct derivative-based algorithm as in Section 4. During the simulations, we test the circle maneuvers with different period T, and the results are given in Figures 3, 4, and 5.

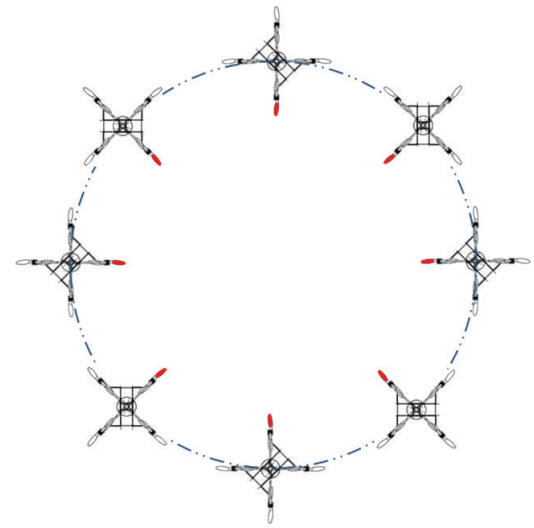

Figure 2: Circle maneuver trajectory.

From these figures, we can obtain the following results.

(1) For input constraints (Figures 3 and 5), only the maneuvering trajectories with $T = 10$ s are infeasible because the required dragging force is almost 210 N, larger than the maximum permitted value of 190.8 N, while the other inputs, that is, the three moments, are all much smaller than the maximum values.

(2) For angular velocity constrains (Figure 5), again, only the maneuvering trajectories with $T = 10$ s are infeasible, and the maneuvering trajectories with $T = 12.5$ s are *dangerous* since the maximum angular velocity approaches the maximum value of the permitted angular velocity.

(3) Based on the attitude constrains (Figure 4), the maneuvering trajectories with $T = 14.3$ s, 12.5 s, and 10 s are all infeasible.

Finally, from the preceding results, it can be seen that the circle maneuvers are really a highly dynamical maneuver, and the quantitative analysis prior to the flight test is useful and necessary to evaluate whether or not a maneuver trajectory is feasible or even dangerous.

5. Conclusions

In this paper, the flight maneuvering trajectories evaluation problem is researched, and the algorithm differentiation (AD) is used to realize the inverse computation of the rotor-flying robot (RFR) system. This paper starts from constructing nonlinear dynamical model of an RFR system. Then, we show that the RFR system model is flat taking translational positions and yaw angle as flat outputs. After that, the scheme of AD based inverse computation is introduced, that is,

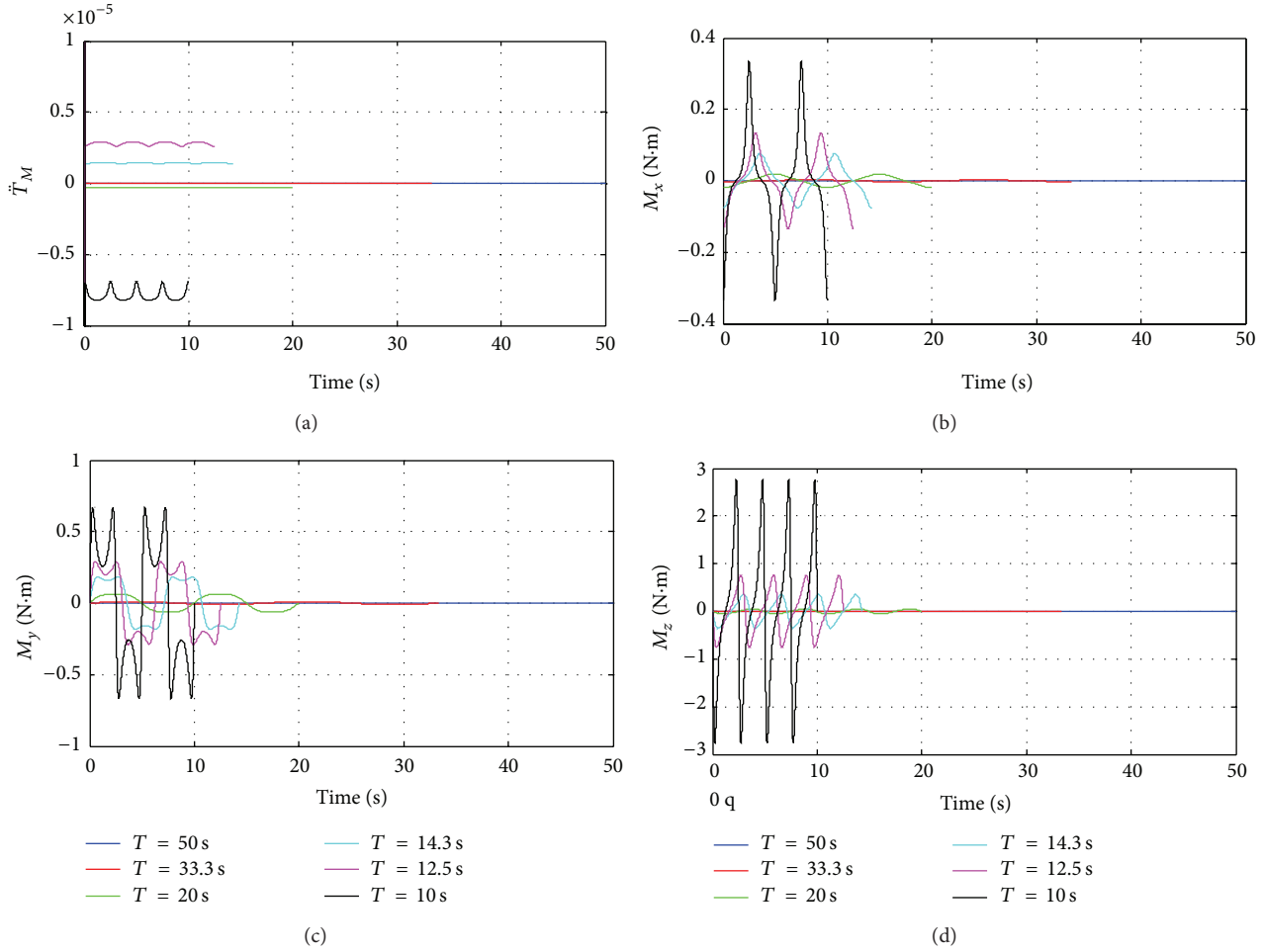

Figure 3: Control input: $\ddot{T}_M$ (a); M_x (b); M_y (c); M_z (d).

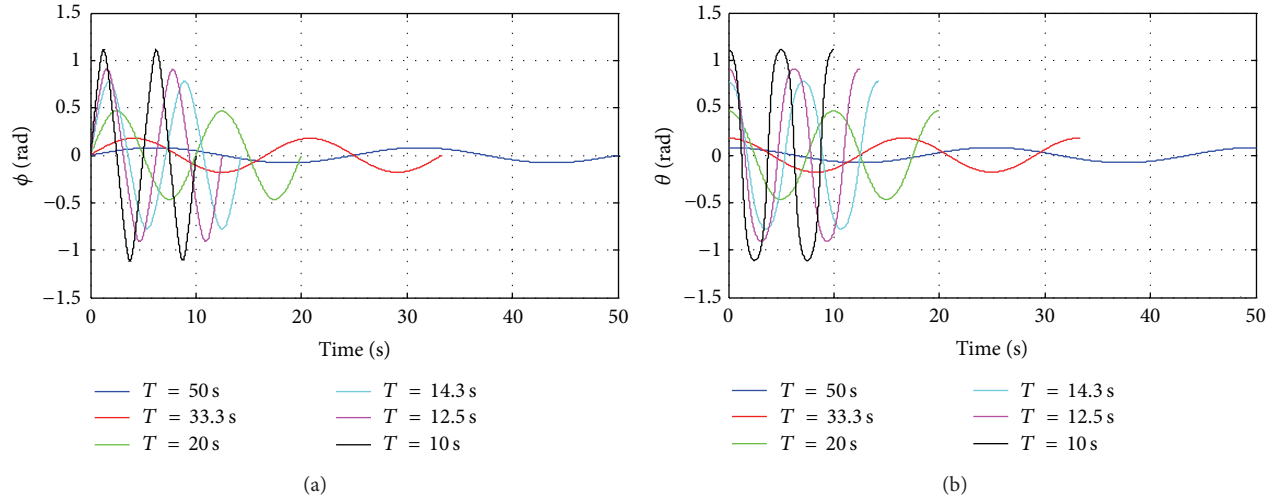

Figure 4: State: ϕ (a); θ (b).

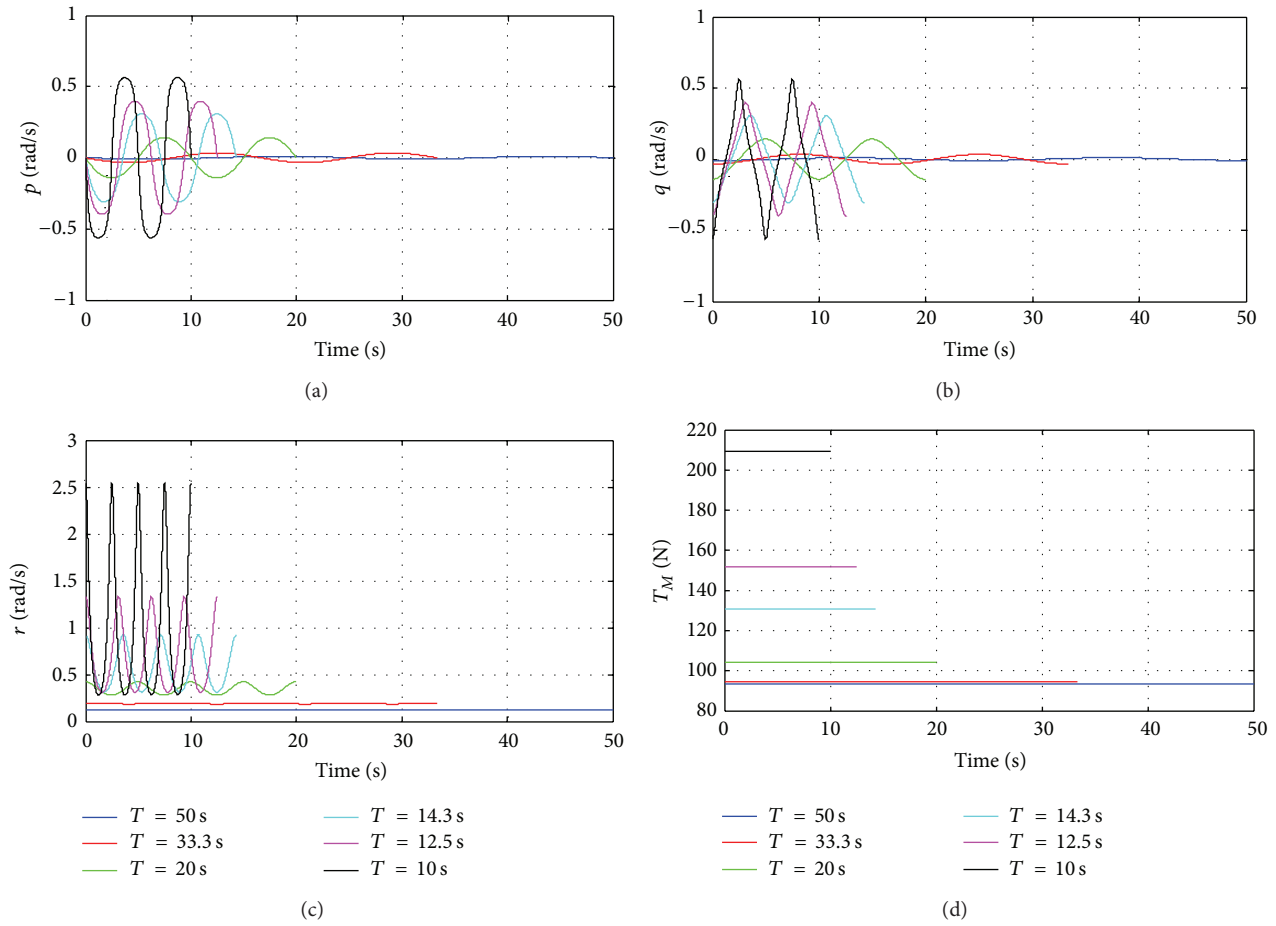

Figure 5: State: p (a); q (b); r (c); T_M (d).

computing the internal states and inputs based on the desired flat outputs. Finally, with the proposed scheme, a typical flight maneuver, that is, the circle maneuver, is analyzed.

The advantages of our proposed algorithm are as follows: (1) only desired positions and their derivatives, which is the case for many planning algorithm, and the feasibility of the trajectories (including inner states and inputs) can be evaluated; (2) inverse computation can be very accurate because AD is a numerical algorithm that can precisely obtain the derivative of a nonlinear function. Thus, an accurate pointwise numerical feasibility analysis of planned trajectories becomes possible.

The method proposed in this paper can be used to not only evaluate the feasibility and validity of the maneuver flying trajectories, but also realize the tracking control algorithm, where the computed internal states and inputs are useful for attenuating the online computational burden of the tracking controller and thus improving the closed loop performance. This will be one of our future's works.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to present their thanks to Professor Klaus Röbenack at Technique University of Dresden (Germany) for the related discussion and suggestions on AD algorithm. This work is supported by the National Natural Science Foundation of China (Grants no. 61035005 and no. 61203340).

References

[1] Y. Q. He and J. D. Han, "Acceleration-feedback-enhanced robust control of an unmanned helicopter," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 4, pp. 1236–1250, 2010.

[2] J. H. Gillula, G. M. Hoffmann, M. P. Vitus, and C. J. Tomlin, "Applications of hybrid reachability analysis to robotic aerial vehicles," *International Journal of Robotics Research*, vol. 30, no. 3, pp. 335–354, 2011.

[3] F. Andert, F. Adolf, L. Goormann, and J. Dittrich, "Mapping and path planning in complex environments: an obstacle avoidance approach for an unmanned helicopter," in *Proceedings of the IE E E International Conference on Robotics and Automation*, pp. 745–750, Shanghai, China, May 2011.

[4] G. Flores, S. T. Zhou, R. Lozano, and P. Castillo, "A vision and GPS-based real time trajectory planning for a MAV in unknown

and low-sunlight environments," *Journal of Intelligent Robotic Systems*, vol. 74, pp. 59–67, 2014.

[5] P. Bristeau, E. Dorveaux, D. Vissière, and N. Petit, "Hardware and software architecture for state estimation on an experimental low-cost small-scaled helicopter," *Control Engineering Practice*, vol. 18, no. 7, pp. 733–746, 2010.

[6] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. E. Johnson, A. Ansar, and L. Matthies, "Vision-aided inertial navigation for spacecraft entry, descent, and landing," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 264–280, 2009.

[7] M. Bangura and R. Mahony, "Nonlinear dynamic modeling for high performance control of a quadrotor," in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA '12)*, pp. 1–10, Wellington, New Zealand, December 2012.

[8] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Precision flight control for a multi-vehicle quadrotor helicopter testbed," *Control Engineering Practice*, vol. 19, no. 9, pp. 1023–1036, 2011.

[9] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '13)*, pp. 3480–3486, Tokyo, Japan, 2013.

[10] D. Song, J. Han, and G. Liu, "Active model-based predictive control and experimental investigation on unmanned helicopters in full flight envelope," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1502–1509, 2013.

[11] O. Purwin and R. D'Andrea, "Performing aggressive maneuvers using iterative learning control," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1731–1736, Kobe, Japan, May 2009.

[12] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '10)*, pp. 1642–1648, Anchorage, Alaska, USA, May 2010.

[13] K. Röbenack and O. Vogel, "Computation of state and input trajectories for flat systems using automatic differentiation," *Automatica*, vol. 40, no. 3, pp. 459–464, 2004.

[14] G. Andreas and W. Andrea, *Evaluating Derivatives: Principles and Techniques of Algorithm Differentiation*, Society for Industrial and Applied Mathematics, Philadelphia, Pa, USA, 2nd edition, 2008.

[15] B. Christianson, "Reverse accumulation and accurate rounding error estimates for Taylor series coefficients," *Optimization Methods and Software*, vol. 1, no. 1, pp. 81–94, 1992.

[16] K. Röbenack, "Automatic differentiation and nonlinear controller design by exact linearization," *Future Generation Computer Systems*, vol. 21, no. 8, pp. 1372–1379, 2005.

[17] K. Röbenack, J. Winkler, and S. Q. Wang, "LIEDRIVERS—a toolbox for the efficient computation of Lie derivatives based on the object-oriented algorithmic differentiation package ADOL-C," in *Proceedings of the 4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, ETH Zurich, Zurich, Switzerland, 2011.