



# A numerical approach for attitude control of a quadrotor

Huu-Phuc Nguyen, Jérôme de Miras, Ali Charara, Stéphane Bonnet

## ► To cite this version:

Huu-Phuc Nguyen, Jérôme de Miras, Ali Charara, Stéphane Bonnet. A numerical approach for attitude control of a quadrotor. 9th International Micro Air Vehicle Conference and Flight Competition (IMAV 2017), Sep 2017, Toulouse, France. pp.23-28. hal-01638197

**HAL Id: hal-01638197**

**<https://hal.science/hal-01638197>**

Submitted on 20 Nov 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A numerical approach for attitude control of a quadrotor

Huu-Phuc Nguyen, Jérôme De Miras\*, Ali Charara and Stéphane Bonnet  
Sorbonne universités, Université de Technologie de Compiègne,  
CNRS, Heudiasyc UMR 7253, CS 60319, 60203 Compiègne CEDEX

## ABSTRACT

This paper deals with a numerical approach to control aggressive maneuvers of a multi-rotor aerial vehicle. The proposed controller uses an approximate tabulated one-step time discretization of the state-space model to find out the outputs of controller. Its objective is to minimize the distance between the plant output and a linear well chosen closed loop system used as reference, leading the system to adopt its dynamical behavior. The prediction horizon is only one step time that ensures the execution time is completely bounded. The results from simulation for quadrotor show the performance and robustness of the proposed controller.

## 1 INTRODUCTION

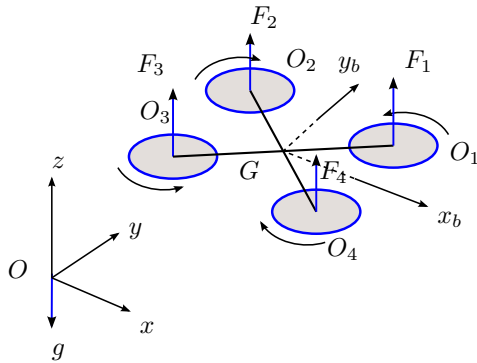


Figure 1: Quadrotor model

A quadrotor is a flying vehicle with four rotors. Figure 1 presents a scheme of the system in a 3D space. The rotor  $i$  rotates at speed  $\omega_i$  and generates a lift force  $F_i$  and a drag torque. The quadrotor is an under-actuated system with four inputs and six degrees of freedom. The control problem for the quadrotor is usually divided into two stages: the attitude tracking control and the stabilization of the position. Usually, the position controller generates a desired attitude for the attitude controller. To provide solutions, a lot of control techniques were used: classic PID controllers, adaptive controllers, predictive controllers, controllers based on the Lyapunov criterion, etc.

Looking at the challenge of control of the quadrotor in the case of aggressive maneuvers in the literature, the following works contain many interesting points for studying and

comparing. An open loop control was proposed in [1] using reachable sets where the complex aerobatic flights were decomposed into sequences of discrete maneuvers. A trajectory generation for multiple-flips has been proposed in the work of [2] using a simple learning approach. A survey of the methods for attitude control of a rigid body can be found in [3]. The minimal time problem has been considered in the work of [4]. In [5], a quadrotor with flip back behavior was used; the trajectory generation was solved in order to apply a state feedback controller based on a Lyapunov function. A full quaternion based attitude control for quadrotor was introduced in [6]. The authors in [7] designed a controller for flip control on Lie group  $SE(3)$ . A learning control used in [8] allows the quadrotor doing aggressive maneuvers. The work in [9] have applied an optimal control LQR.

In this context, this paper proposes an attitude control for a quadrotor using a numerical approach. This controller uses a tabulated numerical model to represent the dynamics of the system as a prediction map over one sample time step. Based on this prediction map, the inputs of the system will be calculated upon the admissible input space as a solution of the minimization problem of the difference between the desired output and the predicted output. The rest of the paper will be organized as follows: firstly, the next section deals with the quadrotor dynamics model, then the scheme of the proposed control will be introduced as its application for attitude control of the quadrotor. Finally some results from MATLAB/SIMULINK will be presented.

## 2 QUADROTOR MODEL

### 2.1 Mathematical model

In the following, vectors and matrices are denoted in bold font. The quadrotor depicted on the Figure 1 is operated by changing the speeds of its four rotors. Using the equation of Newton-Euler, the system is given by the following equations in the body frame  $Gx_b y_b z_b$ :

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{v} \\ \dot{\mathbf{v}} = f\mathbf{R}(q)\mathbf{e}_3 - mg\mathbf{e}_3 \\ \dot{q} = \frac{1}{2}\Xi(q)\Omega \\ \mathbf{J}\dot{\Omega} = (\mathbf{J}\Omega) \times \Omega + \Gamma \end{cases} \quad (1)$$

with  $q = [q_0, q_1, q_2, q_3]^T$  a unit quaternion and

$$\Xi(q) = \begin{bmatrix} -q_1, q_0, -q_3, q_2 \\ -q_2, q_3, q_0, -q_1 \\ -q_3, -q_2, q_1, q_0 \end{bmatrix}^T$$

\*Email address(es): (huu-phuc.nguyen,demiras,ali.charara,bonnetst)@hds.utc.fr

The variables and the parameters of the quadrotor are described in the following table:

|                                  |                                 |
|----------------------------------|---------------------------------|
| $G$                              | Center of mass                  |
| $\mathbf{x} = [x, y, z]^T$       | Position of quadrotor           |
| $\mathbf{v} = [V_x, V_y, V_z]^T$ | Velocity of the quadrotor       |
| $m$                              | Mass of the quadrotor           |
| $\mathbf{J}$                     | Inertia of the quadrotor        |
| $\omega_i$                       | i-rotor's speed                 |
| $\omega_{max}$                   | Maximum speed of the rotors     |
| $\Omega$                         | Angular velocity                |
| $g$                              | Gravity                         |
| $\mathbf{R}(q)$                  | Rotation matrix                 |
| $\mathbf{e}_3 = [0, 0, 1]^T$     | Unit vector                     |
| $k_t$                            | Thrust coefficient              |
| $k_c$                            | Drag torque coefficient         |
| $L$                              | Half-distance between $F_1 F_2$ |

Table 1: Notations

The thrust force and the moments applied on the main body are calculated with the speeds of the rotors as follows:

$$\begin{bmatrix} f \\ \Gamma_x \\ \Gamma_y \\ \Gamma_z \end{bmatrix} = \begin{bmatrix} k_t & k_t & k_t & k_t \\ k_t L & k_t L & -k_t L & -k_t L \\ -k_t L & k_t L & k_t L & -k_t L \\ -k_c & k_c & -k_c & k_c \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \quad (2)$$

The skew-matrix operation of a vector  $\mathbf{x} = [x_1, x_2, x_3]^T$  represents an easy way to compute the cross product of two vectors,  $\mathbf{x} \times \mathbf{y} = [\mathbf{x}]^\times \mathbf{y}$ .

$$[\mathbf{x}]^\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

The inverse operation is denoted as *vee*-operation such as if  $[\mathbf{x}]^\times = \mathbf{X}$  then  $\mathbf{X}^\vee = \mathbf{x}$ . Considering only the attitude of the quadrotor and the vertical dynamic, the equation above can be written as (3).

$$\begin{cases} \dot{v}_z = f\mathbf{R}(9) - mg \\ \dot{\mathbf{R}} = \mathbf{R}[\Omega]^\times \\ \mathbf{J}\dot{\Omega} = (\mathbf{J}\Omega) \times \Omega + \Gamma \end{cases} \quad (3)$$

where  $\mathbf{R}(9)$  is the last item of the rotation matrix  $\mathbf{R}$  and  $\Gamma = [\Gamma_x, \Gamma_y, \Gamma_z]^T$ . The system becomes a full actuated system with four inputs and four degrees of freedom.

## 2.2 Full quadrotor model

There are many toolkits and drag-drop environments to model the dynamics a vehicle using the multibody dynamics theory, for example SimMechanics from Mathworks, MapleSim from MapleSoft. In this paper, we used a quadrotor Parrot-AR2 model built with MapleSim depicted on Figure 2. This model contains a main rigid body, four arms with four brushless motors integrating four propellers to generate

the aerodynamics forces. This model also includes some virtual sensors to obtain the quadrotor's state: the position, the velocity, the orientation and the angular velocity. This model is exported to SIMULINK's environment.

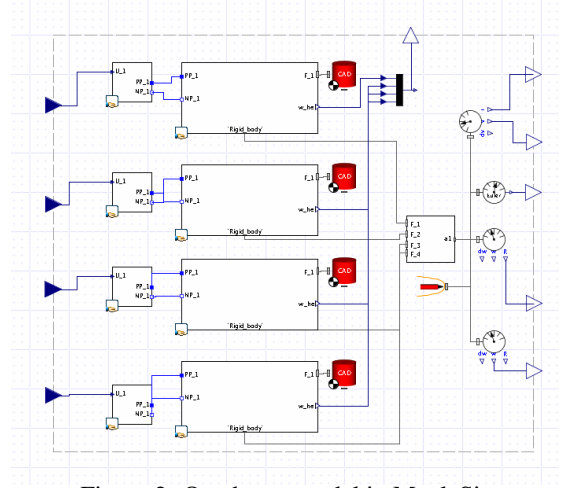


Figure 2: Quadrotor model in MapleSim

The parameters of the quadrotor are given in Table 2.

| Parameters     | Value                 | Unit         |
|----------------|-----------------------|--------------|
| $m$            | 0.506                 | $kg$         |
| $I_x$          | $2.38 \times 10^{-3}$ | $kg.m^2$     |
| $I_y$          | $3.85 \times 10^{-3}$ | $kg.m^2$     |
| $I_z$          | $5.9 \times 10^{-3}$  | $kg.m^2$     |
| $L$            | 0.15                  | $m$          |
| $k_t$          | $2.3 \times 10^{-5}$  |              |
| $k_c$          | $2 \times 10^{-6}$    |              |
| $\omega_{max}$ | 350                   | $rad.s^{-1}$ |

Table 2: Quadrotor parameters

## 3 PROPOSED CONTROL SCHEME

### 3.1 Online control algorithm

The time-invariant model of the system has the following form:

$$\begin{cases} \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) = h(\mathbf{x}(t)) \end{cases} \quad (4)$$

where  $\mathbf{x} \in \mathcal{S} \subset \mathbb{R}^n$  is the system state vector,  $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^d$  its input control vector, and  $\mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^m$  its output vector. The discretized form of the system is needed in order to construct a discrete-time control law. Generally, it is difficult to obtain an exact analytic expression for the discretization of the system. However, the numerical integration of  $f$  gives its approximate value for a given state and input vector as the form:

$$\begin{cases} \mathbf{x}_{k+1} = p_x(\mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{y}_{k+1} = p_y(\mathbf{x}_k, \mathbf{u}_k) \end{cases} \quad (5)$$

As mentioned before, the expression for  $p$  is unknown then the minimization problem (9) has to be solved using a

derivative free optimization solver. In this subsection an iterative algorithm will be used to approximate the solution of (9). This algorithm starts by using the input interval  $\mathcal{U} = [\underline{\mathbf{u}}, \bar{\mathbf{u}}]$  that is used to build the prediction map.

$$\mathcal{U}_{i \in 1, \dots, d} = \left\{ u_j | u_j = \underline{u}_i + j \frac{\bar{u}_i - \underline{u}_i}{N_i}, j \in 0, \dots, N_i \right\}$$

The input space is divided using triangularization technique that is based on the simplex and the affine-envelope of a set of vector. These simplexes are also used at the interpolation step to calculate the barycentric coordinates of a point that is not belong to the rectangular grid  $\mathcal{G}$ . In this algorithm, the simplexes and the affine-envelope are defined as:

$$\mathbf{conv}(\mathcal{X}) = \left\{ \sum_{i=1}^k \omega_i \cdot \mathbf{x}_i | \mathbf{x}_i \in \mathcal{X}, \omega_i \in \mathbb{R}^+, \sum_{i=1}^k \omega_i = 1 \right\}$$

$$\mathbf{aff}(\mathcal{X}) = \left\{ \sum_{i=1}^k \omega_i \cdot \mathbf{x}_i | \mathbf{x}_i \in \mathcal{X}, \omega_i \in \mathbb{R}, \sum_{i=1}^k \omega_i = 1 \right\}$$

The algorithm based upon the work presented in [11], [13] is illustrated on Figure 4. The idea of this algorithm is to find out a simplex  $\sigma_f$  in the inputs space containing the solution of the problem (9) that minimizes the distance of the corresponding simplex  $\Gamma_f$  in the output space to the reference point  $\mathbf{y}^g$ . Firstly, one input simplex  $\sigma_0$  is chosen permitting to calculate its output simplex  $\Gamma_0$ . A quasi-gradient is then built based on the projection of  $\mathbf{y}^g$  onto  $\Gamma_0$ . The simplex  $\sigma_0$  is reflected about its edge satisfying the output simplex moves toward the reference point  $\mathbf{y}^g$  while keeping its beside the input space. This procedure is repeated until the final simplex  $\sigma_f$  is found out. Finally, the control input is calculated as an orthogonal projection of this point onto the solution simplex  $\sigma_f$ . To compensate the error of the prediction map  $\varepsilon_k$ , the output simplex  $\Gamma$  is calculated by  $\Gamma = \mathbf{P}_y \cdot p(\sigma, \mathbf{u}) - \varepsilon_k$ .

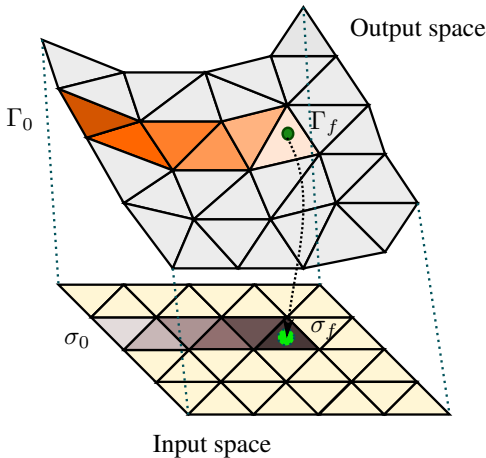


Figure 4: Optimal algorithm visualization

### 3.3 Implementation of the algorithm for the quadrotor

For the implementation of the algorithm, the prediction map should be built from the equation dynamics of the system. The reference equation is (3). The difficulty is that the orientation represented by the unit quaternion contains a constraint. The norm of the quaternion is equal to one  $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$ , so that, it is not possible to use directly  $q$  as four independent elements. In the other hand, the use of the three Euler angles contains a singularity. Avoiding these problems, a new variable will be introduced in order to represent the orientation dynamics. Define now the rotation error between two steps as

$$\Delta \mathbf{R}_{k+1} = \frac{1}{2} (\mathbf{R}_{k+1}^T \mathbf{R}_k - \mathbf{R}_k^T \mathbf{R}_{k+1}) \quad (11)$$

Note that the rotation error matrix  $\Delta \mathbf{R}_{k+1}$  can be verified being a skew-matrix. So that, this matrix is represented by using three independent variables  $\mathbf{e}_R^{k+1} = \Delta \mathbf{R}_{k+1}^\vee$ . Then from (3), the sub-dynamics of the quadrotor can be written in a discrete time domain as (12).

$$[v_z^{k+1}, \mathbf{e}_R^{k+1}, \Omega_{k+1}]^T = F(f_k, \Gamma_k, v_z^k, \mathbf{e}_R^k, \Omega_k, \mathbf{R}_k(9)) \quad (12)$$

Recall that the rotation matrix with the angular velocity  $\Omega_k$  and the sampling time  $T_s$  is calculated by

$$\mathbf{R}_{k+1} = \mathbf{R}_k \exp(T_s \Omega_k) \quad (13)$$

Thanks to the Rodrigues' rotation formula,  $\Delta \mathbf{R}_{k+1}$  has the approximation form:

$$\Delta \mathbf{R}_{k+1} \approx -T_s [\Omega_k]^\times \quad (14)$$

That gives the rotation error approximated by

$$\mathbf{e}_R^{k+1} \approx -T_s \Omega_k \quad (15)$$

Combination of (15), (12) and (3) allows a method to build an approximation map of the attitude dynamics of the quadrotor. Once the prediction map is built, the control algorithm above will be used for the stabilization of the attitude of the quadrotor.

## 4 SIMULATION RESULTS

In this section, we will introduce some simulation results from MATLAB/SIMULINK. The quadrotor model used in these simulations has been exported from MAPLESIM. The sampling time  $T_s = 0.01s$  was chosen for the prediction map. A 100 Hz frequency for the control of a quadrotor is rather a low value. The internal reference linear state-space system for each axis  $\mathbf{e}_R$ ,  $\Omega$  and  $v_z$  has an equivalent canonical first-order transfer function  $W(s) = \frac{\tau_s}{\tau_s s + 1}$  with a time constant of  $\tau_s = 30^{-1} s$ ,  $\tau_s = 50^{-1} s$  and  $\tau_s = 2^{-1} s$  respectively. In the first simulation, the attitude trajectory is constructed from three RPY angles (roll, pitch, yaw)  $y_{di} = 0.5 \sin(t - i)$

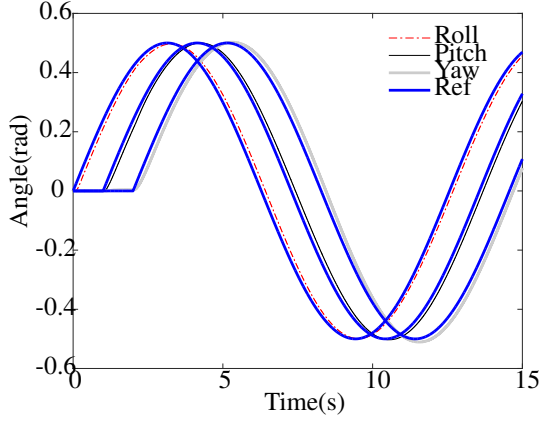


Figure 5: RPY angles and its reference

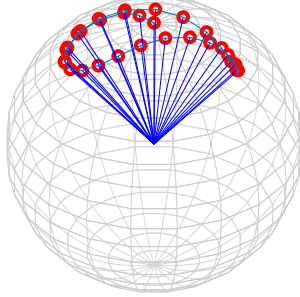


Figure 6: Attitude tracking

with  $i = 1, 2, 3$  for roll-pitch-yaw angle respectively. The response angles from the quadrotor and its reference are shown in Figure 5.

This figure shows that the quadrotor tracked well the desired attitude. On the illustration shown in Figure 6, the trajectory generated by the vector  $\mathbf{e}_3 = [0, 0, 1]^T$  converges to its reference.

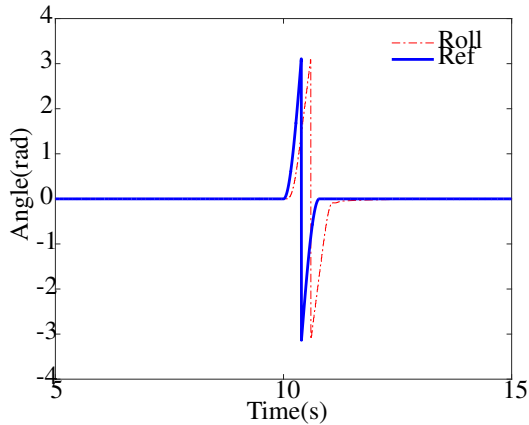


Figure 7: Roll angle and its reference

The second test is performed by doing a single flip about the  $x$ -axis. The pitch and yaw angle are driven to zeros while the roll angle was tracking a rotating trajectory. In this test, the desired vertical velocity  $v_z$  is controlled so as to keep the

position of the quadrotor at 2.5 m. Figure 8 shows that the vertical unit vector  $z_b$  rotates well one perfect round about  $x$ -axis. This significant is also justified in Figure 7.

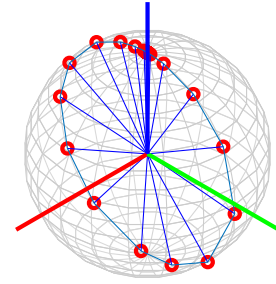
Figure 8: Rotation about  $x$ -axis

Figure 9 shows the vertical position  $z$ , its reference and the vertical velocity. The position  $z$  (dash-dotted curve) tends to its reference 2.5 m. The descent while flipping explains why the user usually accelerate the quadrotor before and during doing flips.

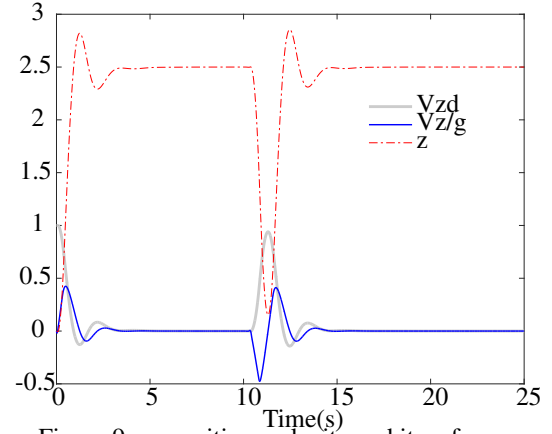
Figure 9:  $z$ -position, velocity and its reference

Figure 10 shows the rotors' nominal speed  $\omega_i/\omega_{max}$ . In hover flight, the rotors' speed are kept at a constant value. To do a single flip, Figure 10 shows the left rotors' speed  $\omega_1$  and  $\omega_2$  are increased while the right rotors' speed  $\omega_3$  and  $\omega_4$  are decreased in order to generate a torque that twists the quadrotor about  $x$ -axis. After the loop is perfectly done, the controller has gotten the smallest speed for the rotors to stabilize faster the roll angle. Finally, the vertical position is compensated.

Figure 11 shows the nominal inputs  $T = f/(k_t * \omega_{max}^2)$ ,  $G_1 = \Gamma_x/(k_t * L * \omega_{max}^2)$ ,  $G_2 = \Gamma_y/(k_t * L * \omega_{max}^2)$  et  $G_3 = \Gamma_z/(k_c * \omega_{max}^2)$ . The nominal thrust  $T$  also shows that the acceleration was increased to accelerate the quadrotor in order to perform the looping while the roll torque  $G_1$  rotates the system about  $x$ -axis.

## 5 CONCLUSION

This paper presents a numerical tabulated approach to the attitude control problem. The numerical behavior of the sys-

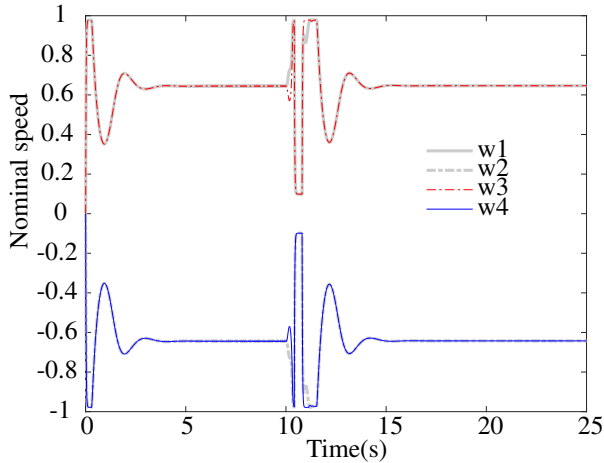


Figure 10: Rotors' nominal speed

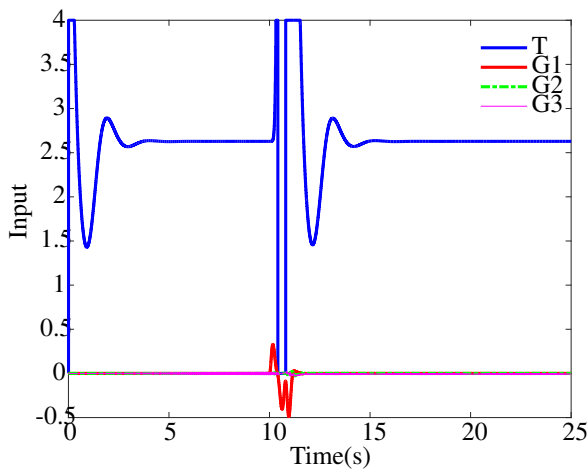


Figure 11: Looping: input

tem is a hybrid model that uses the incremental error between two time steps, avoiding the singularity and ambiguity of the attitude representation. This control could be combined with a position control in order to track complex position trajectory. The validation of the proposed controller on an embedded system requires a good observer for the attitude. Moreover, the prediction map could be improved online using learning-based method.

## REFERENCES

- [1] J. H. Gillula, H. Huang, M. P. Vitus, and C. J. Tomlin. Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1649–1654, 2010.
- [2] S. Lupashin, A. Schollig, M. Sherback, and R. D'Andrea. A simple learning strategy for high-speed quadcopter multi-flips. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1642–1648, 2010.
- [3] N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch. Rigid-body attitude control. *IEEE Control Systems*, 31(3):30–51, 2011.
- [4] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525, 2011.
- [5] Cutler Mark and How Jonathan. *Actuator Constrained Trajectory Generation and Control for Variable-Pitch Quadrotors*. Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, 2012.
- [6] E. Fresk and G. Nikolakopoulos. Full quaternion based attitude control for a quadrotor. In *Control Conference (ECC), 2013 European*, pages 3864–3869, 2013.
- [7] F. Goodarzi, D. Lee, and T. Lee. Geometric nonlinear pid control of a quadrotor uav on  $se(3)$ . In *Control Conference (ECC), 2013 European*, pages 3845–3850, 2013.
- [8] T. Tomix, M. Maier, and S. Haddadin. Learning quadrotor maneuvers from optimal control and generalizing in real-time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1747–1754, 2014.
- [9] Sara Spedicato, Giuseppe Notarstefano, Heinrich H. Bulthoff, and Antonio Franchi. *Aggressive Maneuver Regulation of a Quadrotor UAV*, pages 95–112. Springer International Publishing, Cham, 2016.
- [10] R. Munos and A. Moore. Barycentric interpolators for continuous space & time reinforcement learning. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 1024–1030, Cambridge, MA, USA, 1999. MIT Press.
- [11] Stéphane Bonnet. *Approches numérique pour la commande des systèmes dynamiques*. PhD thesis, Université de Technologie de Compiègne, France, 2008.
- [12] J. De Miras and S. Bonnet. Nonlinear control of a magnetic levitation shaft by numerical inversion of its behavioral model. In *13th IEEE European Control Conference*, Strasbourg, France, June 2014.
- [13] H-P. Nguyen, J. De Miras, S. Bonnet, and A. Charara. Nonlinear control of the pvtol aircraft by numerical inversion of its behavioral model. In *23th IEEE International Conference on Control Application*, Antibes, France, October 2014.