SOFTWARE IN THE LOOP SIMULATION FOR MULTIROTOR POSITION TRACKING USING A LINEAR CONSTRAINED MODEL PREDICTIVE CONTROLLER

Igor Afonso Acampora Prado*, Davi Ferreira de Castro*, Pedro Filizola Sousa Maia Gonçalves*, Davi Antônio dos Santos*

*Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
Instituto Tecnológico de Aeronáutica
São José dos Campos, São Paulo, Brasil

Emails: igorap@ita.br, davifc@ita.br, pedrofsm@ita.br, davists@ita.br

Abstract— The multirotor unmanned aerial vehicles (UAVs) are experiencing a currently growing interest due to their high maneuverability, simplified mechanics and vertical takeoff and landing capability. Also for these reasons, the multirotor UAVs are suitable test-bed platforms for different position tracking control techniques. This paper solves the problem of safely controlling the position tracking of a multirotor UAV by using a linear state-space model predictive controller (MPC) formulation. The optimization is performed by replacing the original conic constraint set by a circumscribed pyramidal space that renders a linear set of inequalities on the total thrust vector. In order to improve the design efficiency of such kind of control systems, the present work uses a low-cost software-in-the-loop (SIL) simulation scheme for performance evaluation of control law for quadricopter. The scheme consists of two computers interconnected by an Ethernet network using the User Datagram Protocol (UDP). Basically, one computer runs MATLAB/Simulink while the other runs X-Plane. The control law is implemented in Simulink, whereas the vehicle dynamics as well as the flight environment are simulated in X-Plane.

Keywords— Multirotor, Position Tracking, Model Predictive Control, Software-In-The-Loop.

Resumo— Os veículos aéreos não tripulados (VANTs) do tipo multirrotor estão esfrentando um momento de crescente interesse devido a sua alta manobrabilidade, mecânica simplificada e capacidade de decolagem e pousou vertical. Também por essas razões, os multirrotores são plataformas de testes adequadas para diferentes técnicas de controle de rastreamento de posição. Este artigo resolve o problema de controlar o rastreamento de posição de um multirrotor com segurança, utilizando controle preditivo baseado em modedo linear em espaço de estados (MPC). A otimização é realizada substituindo o conjunto de restrições do tipo cônica original por um espaço de restrições piramidal circunscrito que projeta um conjunto de inequações lineares sobre o vetor total de impuxo. A fim de melhorar a eficiência de projeto de tal tipo de sistema de controle, o presente trabalho usa um esqueme de simulação de baixo custo Software-In-the-Loop (SIL) para avaliação de desempenho da lei de controle de um quadricóptero. O esquema consiste de dois computadores interconectados por uma rede ethernet usando User Datagram Protocol (UDP). Basicamente, um computador executa MATLAB/Simulink enquanto o outro executa o X-Plane. A lei de controle é implementada em Simulink, enquanto a dinâmica do veículo e o ambiente de vôo, são simulados em X-Plane.

Palavras-chave— Multirrotor, Rastreamento de Posição, Controle Preditivo Baseado em Modelo, *Software-In-The-Loop*.

1 Introduction

The multirotor-type unmanned aerial vehicle (UAV) technology has attracted a great deal of interest of the academia and industry and, consequently experienced a rapid improvement. This interest is justified by features such as their simplified mechanics, low cost, high maneuverability, and vertical take-off and landing (VTOL) capability. Multirotor UAVs has the potential to be employed in applications too risky to human beings or simply intended to increase the efficiency of certain tasks. Examples of applications are building exploration, traffic monitoring, mapping of agricultural areas, search and rescue (Hoffman et al., 2008).

The multirotors have six degrees of freedom (DOF): three of translational motion and three others corresponding to the rotation. However, they feature only four independent control variables: three torque components and the magnitude of the total thrust vector. Therefore, the first difficulty in designing a control system for them could seem to be the need for facing this underactuation characteristic. However, it is necessary to consider that in most of the applications with a market potential, one is in fact concerned to control only four DOFs: the three-dimensional position and the heading angle. The other DOFs must only to be stabilized. A popular control system structure that obviates the dynamics underactuation and allows to control the aforementioned four DOFs is illustrated in Figura 1. The main goal of this scheme is make the true vehicle position $\mathbf{r} \in \mathbb{R}^3$ track some desired position command $\mathbf{r}_\mathrm{d} \in \mathbb{R}^3$. The navigation system is responsible for providing estimated values, in engineering units, of the vehicle attitude $\mathbf{d} \in \mathbb{R}^3$, angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$, position $\mathbf{r}$ and linear velocity $\dot{\mathbf{r}} \in \mathbb{R}^3$. The system structure is divided in two control loops, where the inner loop is responsible for controlling the vehicle attitude and the outer loop carries out

DOI: 10.5540/03.2013.001.01.0049

the trajectory control by providing references of attitude commands $\mathbf{d}_{\text{ref}} \in \mathbb{R}^3$, seeing that the aircraft needs to perform an inclination, in relation to the rotor's plane, to accelerate in a desired direction. Moreover, the outer loop computes the total thrust magnitude $f \in \mathbb{R}$ to control the vertical acceleration. On the other hand, the attitude control block calculates the torque $\boldsymbol{\tau} \in \mathbb{R}^3$ that makes the aircraft follow the attitude commands $\mathbf{d}_{\text{ref}}$. Simulations reduce the development time of

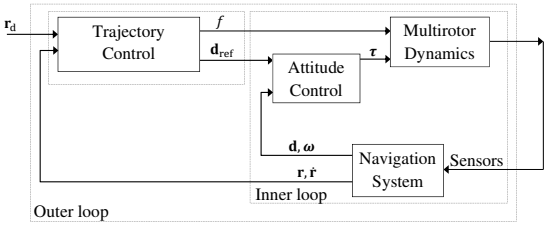

Figure 1: Block diagram structure.

the project, as it allows the testing more quickly and at a lower cost, thus it is possible to compare different control laws in the literature. For the development of control systems, a great choice is the software MATLAB/Simulink®, but is need the mathematical model of the vehicle to perform the simulation. These mathematical models are in some cases much as difficult to be generated or not available (Benini et al., 2011). Furthermore, it is difficult to model mathematically complex robot behaviours in realistic environments. To solve this problem it is utilized a commercially available flight simulator that is able to simulate the vehicle dynamics as well as the flight environment.

This paper refers to a scheme SIL simulation using MATLAB/Simulink® and X-Plane® to face the position tracking problem under constraints on the total thrust vector. The corresponding constraint set is a piece of a cone with an inferior and a superior spherical lids. This problem is solved using a linear state-space MPC, whose optimization is made handy by replacing the original conic constraint set by a circumscribed pyramidal space that renders a linear set of inequalities on the total thrust vector. The paper is organized as follows. Section II describes the multirotor translational dynamic and defines the position tracking problem and its solution. Section III describes Software-In-The-Loop simulation scheme. Section IV simulations tests and Section V contains the conclusion and suggestions for future work.

2 Tracking Position Problem and Solution

This section presents the tracking position problem and its solution taking into account linear constraints set by using MPC.

2.1 Translational dynamic model

Consider the multirotor vehicle and the three Cartesian coordinate systems (CCS) illustrated in Figure 2. Assume that the vehicle has a rigid structure. The body CCS $S_{\text{B}} \triangleq \{X_{\text{B}}, Y_{\text{B}}, Z_{\text{B}}\}$ is fixed to the structure and its origin coincides with the center of mass (CM) of the vehicle. The reference CCS $S_{\text{R}} \triangleq \{X_{\text{R}}, Y_{\text{R}}, Z_{\text{R}}\}$ is Earth-fixed and its origin is at point O. Finally, the CCS $S_{\text{R}'} \triangleq \{X_{\text{R}'}, Y_{\text{R}'}, Z_{\text{R}'}\}$ is defined to be parallel to S_{R}, but its origin is shifted to CM. Assume that S_{R} is an inertial frame.

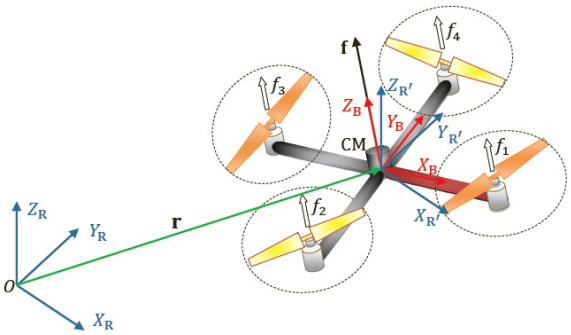

Figure 2: The Cartesian coordinate systems.

Invoking the second Newton's law and neglecting disturbance forces, the translational dynamics of the multirotor illustrated in Figure 2 can be immediately described in S_{R} by the following second order differential equation:

$$\ddot{\mathbf{r}} = \frac{1}{m}\mathbf{f} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}, \tag{1}$$

where $\mathbf{r} \triangleq [r_{\text{x}}\ r_{\text{y}}\ r_{\text{z}}]^{\text{T}} \in \mathbb{R}^3$ is the position vector of CM, $\mathbf{f} \triangleq [f_{\text{x}}\ f_{\text{y}}\ f_{\text{z}}]^{\text{T}} \in \mathbb{R}^3$ is the total thrust vector, m is the mass of the vehicle, and g is the gravitational acceleration. As illustrated in Figure 2, $\mathbf{f}$ is perpendicular to the rotor plane.

Define the inclination angle $\phi \in \mathbb{R}$ of the rotor plane to be the angle between Z_{B} and $Z_{\text{R}'}$. The angle ϕ can thus be expressed by

$$\phi \triangleq \cos^{-1}\frac{f_{\text{z}}}{f}, \tag{2}$$

where $f \triangleq \|\mathbf{f}\|$.

Define the position tracking error $\tilde{\mathbf{r}} \in \mathbb{R}^3$ as

$$\tilde{\mathbf{r}} \triangleq \mathbf{r} - \mathbf{r}_{\text{d}}, \tag{3}$$

where $\mathbf{r}_{\text{d}} \triangleq [r_{\text{d,x}}\ r_{\text{d,y}}\ r_{\text{d,z}}]^{\text{T}} \in \mathbb{R}^3$ is a position command.

Problem 1. Let $\phi_{\max} \in \mathbb{R}$ denote the maximum allowable value of ϕ and $f_{\min} \in \mathbb{R}$ and

$f_{\max} \in \mathbb{R}$ denote, respectively, the minimum and maximum admissible values of f. The problem is to find a control law for $\mathbf{f}$ that minimizes $\tilde{\mathbf{r}}$, subject to the inclination constraint $\phi \leq \phi_{\max}$ and to the force magnitude constraint $f_{\min} \leq f \leq f_{\max}$.

2.2 State-Space Model

Define the state vector $\mathbf{x} \triangleq [r_x \ \dot{r}_x \ r_y \ \dot{r}_y \ r_z \ \dot{r}_z]^{\mathrm{T}} \in \mathbb{R}^6$ and the control input vector $\mathbf{u} \in \mathbb{R}^3$

$$\mathbf{u} \triangleq \frac{1}{m}\mathbf{f} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{4}$$

Let $\mathbf{x}(k) \in \mathbb{R}^6$, $\mathbf{u}(k) \in \mathbb{R}^3$ and $\mathbf{y}(k) \in \mathbb{R}^3$ denote, respectively, the state vector, the control input vector and the controlled output vector all described in discrete-time domain. Using the Euler discretization method with a sample time of $T_s = 30$ ms, the discrete-time linear state-space model is given as

$$\begin{array}{rcl} \mathbf{x}(k+1) & = & \mathbf{A}_{\mathrm{d}}\mathbf{x}(k) + \mathbf{B}_{\mathrm{d}}\mathbf{u}(k) \\ \mathbf{y}(k) & = & \mathbf{C}_{\mathrm{d}}\mathbf{x}(k) \end{array}, \tag{5}$$

where

$$\mathbf{A}_{\mathrm{d}} = \begin{bmatrix} 1 & 0.05 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.05 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.05 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{6\times 6}, \tag{6}$$

$$\mathbf{B}_{\mathrm{d}} = \begin{bmatrix} 0.0013 & 0 & 0 \\ 0.05 & 0 & 0 \\ 0 & 0.0013 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.0013 \\ 0 & 0 & 0.05 \end{bmatrix} \in \mathbb{R}^{6\times 3}, \tag{7}$$

and

$$\mathbf{C}_{\mathrm{d}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 6} \tag{8}$$

2.3 Thrust Vector Constraints

Using (4), the thrust magnitude constraint $f_{\min} \leq f \leq f_{\max}$ can be rewritten in terms of $\mathbf{u}$ as

$$f_{\min} \leq \left\| m\mathbf{u} + m \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \right\| \leq f_{\max}, \tag{9}$$

$$f_{\min} \leq m\sqrt{u_{\mathrm{x}}^2 + u_{\mathrm{y}}^2 + (u_{\mathrm{z}} + g)^2} \leq f_{\max}. \tag{10}$$

Assuming that $0 \leq \phi_{\max} < \pi/2$ rad, the inclination constraint $\phi \leq \phi_{\max}$ can be replaced by

$$\cos\phi \geq \cos\phi_{\max}. \tag{11}$$

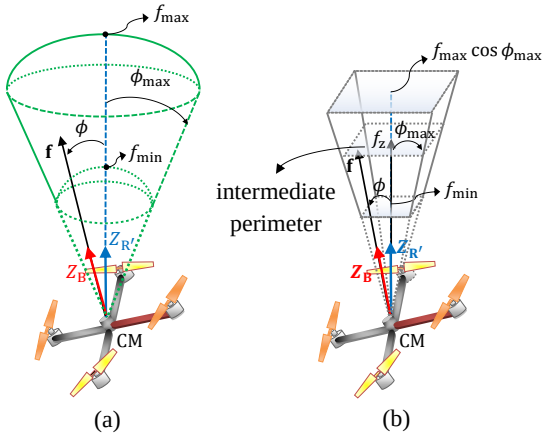

Figure 3: (a) Real constraint space; (b) Linearized constraint space.

Using (2), (11) can be rewritten as

$$\frac{f_{\mathrm{z}}}{f} \geq \cos\phi_{\max}, \tag{12}$$

which in turn, using (4), can be rewritten in terms of $\mathbf{u}$, yielding

$$\frac{u_{\mathrm{z}} + g}{\sqrt{u_{\mathrm{x}}^2 + u_{\mathrm{y}}^2 + (u_{\mathrm{z}} + g)^2}} \geq \cos\phi_{\max}. \tag{13}$$

In short, (10) and (13) give the nonlinear constraints expressed in terms of the components of the control vector $\mathbf{u}$. From the original form of the constraints as defined in Problem 1, one can visualize them as a conic space with spherical inferior and superior lids, as illustrated in Figure 3a. Note that, besides nonlinear, the original constraint space is non-convex.

2.3.1 Linear Constraints

A linear suboptimal constraint space is now obtained as being the pyramidal space circumscribed in the original constraint space of Figure 3a. The new space is depicted in Figure 3b.

From Figure 3b, the new constraint along the $Z_{\mathrm{R}'}$ axis can immediately expressed as

$$f_{\min} \leq f_{\mathrm{z}} \leq f_{\max}\cos\phi_{\max}. \tag{14}$$

Consider an arbitrary f_{z} and let it represent the $Z_{\mathrm{R}'}$ axis coordinate of the dotted perimeter represented in Figure 3b. The corresponding projection on the $X_{\mathrm{R}'} - Y_{\mathrm{R}'}$ plane is given in Figure 4. The later figure defines the distances α and β and enables to expresses them immediately as

$$\alpha = 2\beta\cos\frac{\pi}{4}, \tag{15}$$

and

$$\beta = f_{\mathrm{z}}\tan\phi_{\max}. \tag{16}$$

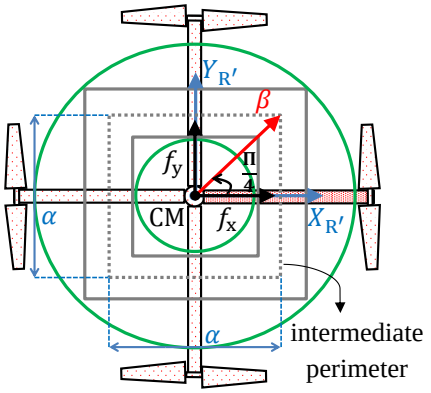

Figure 4: Constraints top view.

Now, replace (16) into (15) to obtain

$$\alpha = \sqrt{2} f_z \tan \phi_{\max} \tag{17}$$

and note that the linear constraints along the horizontal axes are given by $-\alpha/2 \leq f_x, f_y \leq \alpha/2$. To finally obtain in matrix form yields

$$\mathbf{\Lambda}\ \mathbf{f} \leq \boldsymbol{\lambda}, \tag{18}$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} -1 & 0 & -\frac{\sqrt{2}}{2}\tan\phi_{\max} \\ 1 & 0 & -\frac{\sqrt{2}}{2}\tan\phi_{\max} \\ 0 & -1 & -\frac{\sqrt{2}}{2}\tan\phi_{\max} \\ 0 & 1 & -\frac{\sqrt{2}}{2}\tan\phi_{\max} \\ 0 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix} \tag{19}$$

and

$$\boldsymbol{\lambda} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -f_{\min} \\ f_{\max}\cos\phi_{\max} \end{bmatrix}. \tag{20}$$

Finally, using (4), (18) can be immediately reformulate in terms of $\mathbf{u}$ as

$$\bar{\mathbf{\Lambda}}\mathbf{u} \leq \bar{\boldsymbol{\lambda}}, \tag{21}$$

where

$$\bar{\mathbf{\Lambda}} \triangleq m\mathbf{\Lambda} \tag{22}$$

and

$$\bar{\boldsymbol{\lambda}} \triangleq \boldsymbol{\lambda} - m\mathbf{\Lambda}\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{23}$$

2.4 Model Predictive Controller

This paper use the incremental-input formulation. The MPC is responsible for solving a quadratic programming by minimizing a quadratic cost function subject to the aforementioned linear constraints set. Only the first element of the optimized control increments sequence is utilized. The control command that is effectively applied on the plant is obtained by summing the last control input with the optimized control increments sequence The optimization process is repeated at the next sampling instant based on new sensors measurements. The mathematical formulation is found with more details in (Lopes et al., 2011).

2.5 Computing Thrust Magnitude and Attitude Commands

After the optimization process of the control input, it is necessary to convert $\mathbf{u}$ to the total thrust f and the attitude reference angles, which will be utilized as a set-point for some inner attitude control loop. Thus, rewriting (4) as

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = m \begin{bmatrix} u_x \\ u_y \\ u_z + g \end{bmatrix}, \tag{24}$$

Then, by taking the norm of (24), we can define f as

$$f = m\sqrt{u_x^2 + u_y^2 + (u_z + g)^2}. \tag{25}$$

The attitude commands for the inner control loop need to be computed from the vector $\mathbf{f}$. However, it is can be easily seen that there are infinite attitude of S_B to represent the multirotor orientation in S_R, taking in account when the Z_B axis coincides with $\mathbf{f}$. Therefore, it is necessary to specify an unique attitude to the aerial robot. It can be done utilizing the attitude represented by the principal Euler angle/axis (ϕ, $\mathbf{e}$). Define the principal Euler axis $\mathbf{e} \in \mathbb{R}^3$, shown in Figure (5), as

$$\mathbf{e} = \frac{Z_{R'} \times \mathbf{f}_{xy}}{\|Z_{R'} \times \mathbf{f}_{xy}\|}, \tag{26}$$

where $\mathbf{f}_{xy} \triangleq [f_x\ f_y]^T \in \mathbb{R}^2$ denotes the horizontal projection of $\mathbf{f}$.

Utilizing the (2) and (26), it is possible to represent the attitude of S_B in relation to S_R utilizing, for example, Euler angles one obtains ϕ, θ, ψ.

3 SOFTWARE-IN-THE-LOOP SIMULATION SCHEME

The developed scheme provides resources for the evaluation of a trajectory control using a linear constrained MPC applied to multirotors. The schematic simulation contains computers with computing power and graphics to process calculations for the MPC.

DOI: 10.5540/03.2013.001.01.0049

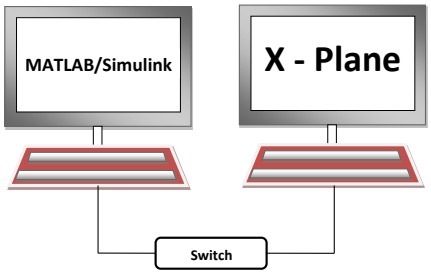

Figure 5: Configuration of network computers.

3.1 Configuration scheme simulation

The scheme consists of two computers connected by a network of Ethernet communication, as shown in Figure 5, where the first computer runs software MATLAB/Simulink® and the second X-Plane®. Control law is implemented in MATLAB/Simulink®, while the dynamics and kinematics of the vehicle and the flight environment are simulated in X-Plane®.

3.2 Simulator X-Plane

For the development of this work was chosen simulator X-Plane® to implement the scheme simulation due to following characteristics: is certified by FAA (Federal Aviation Administration), it is possible to simulate real conditions as the wind and turbulence, the X-Plane® provides a module for the construction of aircraft called PlaneMarker®, Simulation-based blade elements and ease in insertion and extraction of data from the simulator (Indriyanto and Jenie, 2010). In practice, the flight model based on "Blade Element Theory" aerodynamic surfaces of the aircraft (wings, tail surfaces etc.) are split into several parts, the force acting on each of them is calculated by applying a fluid (Benini et al., 2011).

3.3 X-Plane® interfacing with MATLAB/Simulink®

The standard method of X-Plane® to communicate with external processes and machines is through User Datagram Protocol (UDP). The first step to realize the communication between MATLAB/Simulink® and X-Plane® is to define what data will be sent from the X-Plane® for diagrams in MATLAB/Simulink®. Data received by the MATLAB/Simulink® are used to calculate controls, where inputs and outputs data of X-Plane® are illustrated in Figure 6.

Where T_1, T_2, T_3 and T_4 are respectively throttle commands input into the simulated quadricopter in X-Plane. With relation to the data output of the X-Plane® they are defined as follows: r_x, r_y and r_z are the linear positions; $\dot{r}_x, \dot{r}_y$ and $\dot{r}_y$ are the linear velocities; ϕ, θ and ψ are the Euler angles and P, Q e R are the angular velocities.

The sequence of the sending and receiving data by the softwares MATLAB/Simulink® and X-Plane®, can be observed in Figure 3, which appears three-phase flow data.

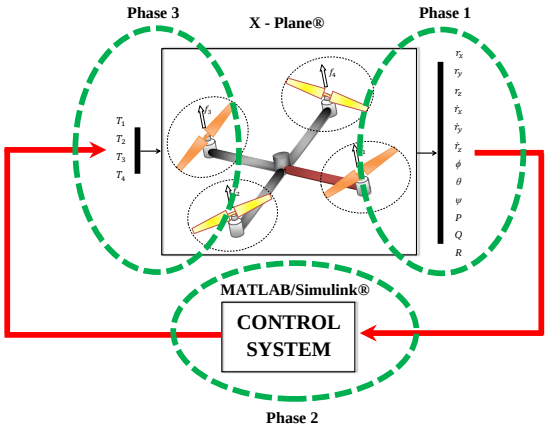

Figure 6: Data flow between X-Plane® and MATLAB/Simulink® control system.

Phase 1: The X-Plane® sends the sensor data via UDP to the aircraft control system implemented in MATLAB/Simulink®. Phase 2: The MATLAB/Simulink® receives the data and performs the calculation of the control law which is then sent to the X-Plane®. Phase 3: The X-Plane® receives control law data via UDP with values for throttle command of the simulated aircraft.

For more details about the communication between MATLAB/Simulink® and X-Plane® work (Figueiredo and Saotome, 2012b) explains with greater resources.

The aircraft model used for simulation was developed on the work of (Figueiredo and Saotome, 2012a)

4 Simulations and Results

All simulations were performed using The X-Plane® and MATLAB/Simulink®. To solve the quadratic programming was utilized the *quadprog* function in Matlab's Optimization ToolBox. The vehicle has mass of m =1 kg. In the whole simulation the MPC parameters were set as $N = 70$ (prediction horizon), $M = 5$ (control horizon), $\phi_{\max} =$ 1 deg, $f_{\max} = 12$ N, $f_{\min} = 2$N $\mu = [1\ 1\ 1]^{\mathrm{T}}$ (output weightings) and $\rho = [0.7\ 0.7\ 0.7]^{\mathrm{T}}$ (control input weightings). The vehicle attitude was stabilized by using a PD controller with $K_{\mathrm{p}} = 0.5$ and $K_{\mathrm{d}} = 0.009$. In order to verify the behavior of the vehicle, a straight position was used as set-point to the aircraft. As shown in Figure 9, the controller was able to make the vehicle follow the reference command with altitude overshoot $M_{\mathrm{p}} = 5\%$. Fig-

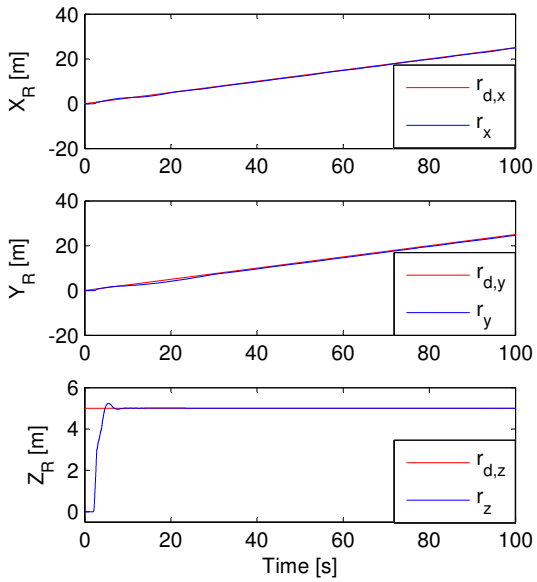

Figure 7: ϕ and f.

ure 10 shows that in the beginning of the simulation the total thrust magnitude was saturated on its upper bound to make the vehicle take off and then stabilized in approximately 9.81N that represents the minimum thrust force to maintain the vehicle on a fixed altitude. Furthermore, the limits to ϕ were respected even with an oscillatory response.

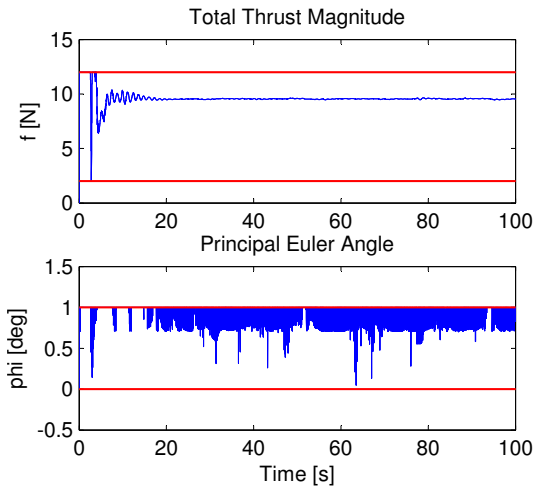

Figure 8: ϕ and f.

5 Conclusions

This paper presents a scheme for Software-In-the-Loop simulation applied to evaluate the performance of position tracking using MPC. The scheme is the integration of two software's. A software responsible for plant simulation and the other to simulate the control system. A strategy was adopted in which the vehicle takes off and follows a straight position. With the predictive control law that was tested in the environment SIL, it has become possible to evaluate the control law tracking position, a more realistic and can be viewed movements of the aircraft in a flight simulator. As future work can expand the position tracking of a single vehicle for multiple vehicles, resulting in a cooperative control. Furthermore, one can carry out an expansion for a simulation Hardware-in-the-Loop (HIL).

Acknowledgments

The authors acknowledge the financial support of FAPEAM - Foundation for Research of the Amazon by means of Master Scholarships, as well as the ITA - Aeronautical Institute of Technology for the support needed to carry out this work.

References

Benini, A., Mancini, A., Minutolo, R., Longhi, S. and Montanari, M. (2011). A modular framework for fast prototyping of cooperative unmanned aerial vehicle, *Journal of Intelligent and Robotic Systems* **65**: 507–520. DOI: 10.1007/s10846-011-9577-1

Figueiredo, H. V. and Saotome, O. (2012a). Modelagem e simulação de veículo aéreo não tripulado (vant) do tipo quadricóptero usando o simulador x-plane e simulink, *Anais do XIX Congresso Brasileiro de Automática, CBA 2012*, Campina Grande, Brazil.

Figueiredo, H. V. and Saotome, O. (2012b). Simulation platform for quadricopter: Using matlab/simulink and x-plane, *XIV Simpósio de Aplicações Operacionais em Áreas de Defesa, SIGE 2012*, São José dos Campos, Brazil.

Hoffman, G. M., Waslander, S. L. and Tomlin, C. J. (2008). Quadrotor helicopter trajectory tracking control, *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii.

Indriyanto, T. and Jenie, Y. I. (2010). Modeling and simulation of a ducted fan unmanned aerial vehicle (uav) using x-plane simulation software, *Regional Conference on Mechanical and Aerospace Technology*, Bali.

Lopes, R. V., Santana, P. H. R. Q. A., Borges, G. A. and Ishihara, J. Y. (2011). Model predictive controle applied to tracking and attitude stabilization of a vtol quadrotor aircraft, *Proceedings of the 21st International Congress of Mechanical Engineering - COBEM*, Natal, Brazil.

