VIRTUAL MODEL OF UNMANNED AERIAL VEHICLE

Juraj Gvoth, Michal Blaho, Tatiana Mudráková

Institute of Robotics and Cybernetics

Faculty of Electrical Engineering and Information Technology

Slovak University of Technology

Ilkovičova 3, 812 19 Bratislava, Slovak Republic

Abstract

Virtual model of the real system is nowadays frequently used tool in the field of process control. It allows to monitor the behavior described by the mathematical formulas that correspond to the real system. Thanks to the virtual models, control algorithms can be verified without damaging the device. MATLAB Simulink environment provides tools for design the mathematical models of the processes whose behavior can be visualized using 3D environment. This paper describes the design of the UAV virtual model (Unmanned Aerial Vehicle), which is often used for the verification of control algorithms - quadcopters (quadrotor helicopter).

1 Introduction

Modeling and simulation are used in the control systems design for many years [1]. They allow control algorithms design without real device attendance, which in many cases can be financially demanding, easy to destroy, eventually unavailable with the required parameters. Models can be found in the learning process very often. These models though may have a demotivational effect on students because there can be seen only ordinary graphs. With the VRML (Virtual Reality Modeling Language) language, there is a opportunity to visualize objects using 3D virtual reality environment [2]. Initially, VRML was deployed on the websites with the 3D scenes in the simple text format. Later, simulation tool MATLAB Simulink used scenes created in VRML language in its own environment. Simulink 3D Animation Extension is able to connect blocks in Simulink with 3D scene and visualize the process [3,4]. It lightened the idea how the system can behave in the real onset [5,6]. This paper describes the quadcopter virtual model design. Based on the mathematical equations, the quadcopter simulation model was developed [7]. Quadcopter was visualized using VRML language and controlled with joystick. This model deployment is able to help in the learning process and in the control algorithms testing, as well.

2 Mathematical model of the quadcopter

The mathematical model describes the quadcopter dynamics in the simplified sense. If we would like to model all the effects on the behavior of quadcopter during the flight, the mathematical model would be considerably more complicated. Simulation will be also complicated, which would then become difficult for a computer to calculate. The mathematical model is taken from the literature [7]. In the Figure 1 the basic structure of quadcopter can be seen and also can be found the world coordinate system, the coordinate quadcopter system, angular velocity directions of each rotor, and the torque and tension forces generated by rotors.

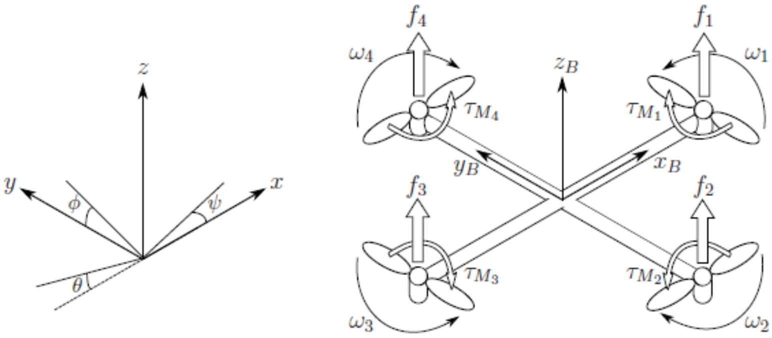

Figure 1: The principal quadcopter view in two coordinate system[7]

The quadcopter position in the world coordinate system is defined by the vector $\boldsymbol{\xi}$ and the vector of rotation angle $\boldsymbol{\eta}$.

$$\boldsymbol{\xi} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

$$\eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{2}$$

Quadcopter displacement in the world coordinate system is defined by the Euler angles, where $\boldsymbol{\phi}$ je roll angle determines the rotation around x-axis, $\boldsymbol{\theta}$ pitch angle represents the rotation of the quadcopter around the y-axis and jaw angle $\boldsymbol{\psi}$ around the z-axis. Vector $\boldsymbol{\nu}$ contains the linear and angular position vectors.

$$\nu = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3}$$

The rotation matrix **R** from the quadcopter coordinate system to the world coordinate system,

$$R = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix} \tag{4}$$

where is $S_x = sin(x)$ and $C_x = cos(x)$. The rotation matrix **R** is orthogonal, thus apply $\mathbf{R}^{-1} = \mathbf{R}^T$ which is the rotation matrix from the world coordinate system to the quadcopter coordinate system. The transformation matrix $\boldsymbol{W_\eta}$ for angular velocities from the world coordinate system to the quadcopter coordinate system and $\boldsymbol{W_\eta^{-1}}$ vice versa.

$$\nu = W_\eta \dot{\eta}$$
$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & C_\theta S_\phi \\ 0 & -S_\phi & C_\theta C_\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{5}$$

$$\dot{\eta} = W_\eta^{-1} \nu$$
$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{6}$$

The quadcopter assume to have symmetric structure with the four rotors in the x-axis and y-axis, the inertia matrix is diagonal matrix **I** where $I_{xx} = I_{yy}$.

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{7}$$

Each of the motors create force $\mathbf{f_i}$ and torque $\boldsymbol{\tau}_{\mathbf{Mi}}$ around the rotor axis,

$$f_i = k\omega_i^2 \tag{8}$$
$$\tau_{M_i} = b\omega_i^2 + I_M \dot{\omega}_i \tag{9}$$

where $\dot{\omega}_i$ is often very small a therefore can be neglected. The sum of forces created by each rotor generates the tension force **T** in the direction of z-axis the quadcopter coordinate system.

$$T = \sum_{i=1}^{4} f_i = k \sum_{i=1}^{4} \omega_i^2$$
$$T_B = \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} \tag{10}$$

The final vector of torques consisting of torques which directions responds to the direction of quadcopter coordinate system,

$$\tau_B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} lk(-\omega_2^2 + \omega_4^2) \\ lk(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^4 \tau_{M_i} \end{bmatrix} \tag{11}$$

where l is the distance between the rotor and the center of mass of the quadcopter and k is the constant for the force created by the rotor. The quadcopter r is assumed to be rigid body and thus Newton-Euler equations can be used to describe its dynamics. at first it will be used equations describing the actual position of the entity in the world coordinate system. In the equation (12) the condition of equilibrium state is recorded in the quadcopter coordinate system, where the force required for the acceleration of mass **m** and the centrifugal force $\mathbf{\nu} \times (\mathrm{m}V_B)$ have to be equal to the gravity $\mathbf{R}^T\mathbf{G}$ and the total thrust of the rotors $\mathbf{T_B}$

$$m\dot{V}_B + \nu \times (mV_B) = R^T G + T_B \tag{12}$$

In the world coordinate system, the centrifugal force is nullified. Thus only the gravitational force and the magnitude and direction of the thrust are contributing in the acceleration of the quadcopter.

But because we need the equation with respect to the world coordinate system, it is necessary to multiply the right side of the equation with rotation matrix from which we get the equation (12).

$$\xi = m\ddot{\xi} = G + RT_B \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{13}$$

After substituting the status description (14) is obtained, which contains the first motion equations of quadcopter

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m} \begin{bmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{bmatrix} \tag{14}$$

In the quadcopter coordinate system the equation (15) is applied, as well. Where the angular acceleration of the inertia $\mathbf{I}\dot{\boldsymbol{\nu}}$, the centripetal forces $\boldsymbol{\nu} \times (\mathbf{I}\boldsymbol{\nu})$ and the gyroscopic forces $\mathbf{\Gamma}$ are equal to the external torque $\boldsymbol{\tau}$.

$$I\dot{\nu} + \nu \times (I\nu) + \Gamma = \tau$$

$$\dot{\nu} = I^{-1}\left(-\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx}p \\ I_{yy}q \\ I_{zz}r \end{bmatrix} - I_r \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega_\Gamma + \tau \right)$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} (I_{yy} - I_{zz})qr/I_{xx} \\ (I_{zz} - I_{xx})pr/I_{yy} \\ (I_{xx} - I_{yy})pq/I_{zz} \end{bmatrix} - I_r \begin{bmatrix} q/I_{xx} \\ -p/I_{yy} \\ 0 \end{bmatrix} \omega_\Gamma + \begin{bmatrix} \tau_\phi/I_{xx} \\ \tau_\theta/I_{xx} \\ \tau_\psi/I_{xx} \end{bmatrix} \tag{15}$$

Equation $\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4$ is applied. In order to work with the rotation angle of quadcopter in the world coordinate system, it is necessary to use the transformation equation (6). To ensure a more realistic model, the member that representing the force exerted by air resistance needs to be add to the equation (14). It represents a diagonal matrix vector multiplication with an aerodynamic drag coefficient in the individual axes and linear velocities of quadcopter in the world coordinate system.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m} \begin{bmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{bmatrix} - \frac{1}{m} \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{16}$$

In a similar way, it is able to add further aerodynamic effects, such as vibration of the propeller blades, or the flow of the air. But this would have resulted in considerably more complex mathematical model.

3 Model verification

Implementation block scheme was designed in MATLAB Simulink environment. Quadcompter constants and coefficients used in the simulation are given in the Table 1.

Table 1: SIMULATION PARAMETERS

Parameter	*Value*	*Unit*
g	9.81	m/s^2
m	0.468	kg
l	0.225	m
k	$2.980*10^{-6}$	-
b	$1.140*10^{-7}$	-
I_M	$3.357*10^{-5}$	$kg\ m^2$
I_{xx}	$4.856*10^{-3}$	$kg\ m^2$
I_{yy}	$4.856*10^{-3}$	$kg\ m^2$
I_{zz}	$8.801*10^{-3}$	$kg\ m^2$
A_x	0.25	kg/s
A_y	0.25	kg/s
A_z	0.25	kg/s

The simulation will issue from the state, when the rotors already have a value of angular velocity that the pulling force generated by them is equal to the gravitational force and thus the quadcopter will be in a stable position. During the first 0.2 seconds speed of all four rotors has been increased by the step change that causes movement of the quadcopter in the positive z direction. Opposite, at the time of 0.2 to 0.4 seconds the speed of all four rotors were reduced and from the time 0.4s were returned to the original value by a step change. Similarly, the angle φ (Roll) changes with fourth rotor speed increase and simultaneously second rotor speed decrease in the time of 0.5 to 0.7 seconds. Increase of angle φ was stopped by the time 0.7 to 0.9 seconds because the fourth rotor speed was reduced and the second rotor speed was increased. This change of the angle will reflects the quadcopter movement in the negative y-axis direction and in the negative z-axis position change direction. To achieve the angle θ (pitch) change, it was necessary in the time of 1 to 1.2 seconds increase the third rotor speed by step response and simultaneously to decrease the second rotor speed. Comparing the previous angle, the increase of the angle was also in this case stopped at the time of 1.2 to 1.4 seconds by the rotor speed decrease. At this point quadcopter starts to move in the positive x-axis direction and it even faster starts to sink on the z-axis. The angle ψ (Yaw) was last modified by increasing the first and third rotor speed and decreasing the second and fourth rotor speed at the time of 1.5 to 7.1 seconds. To prevent this angle not to accumulate, it was also required at the time of 1.5 to 1.7 seconds to reduce the first and third rotor speed and increase the second and fourth rotor speed. The time responses of the rotors speed, position, and rotation angle of the quadcopter body in the global coordinate system are displayed in the Figure 2.

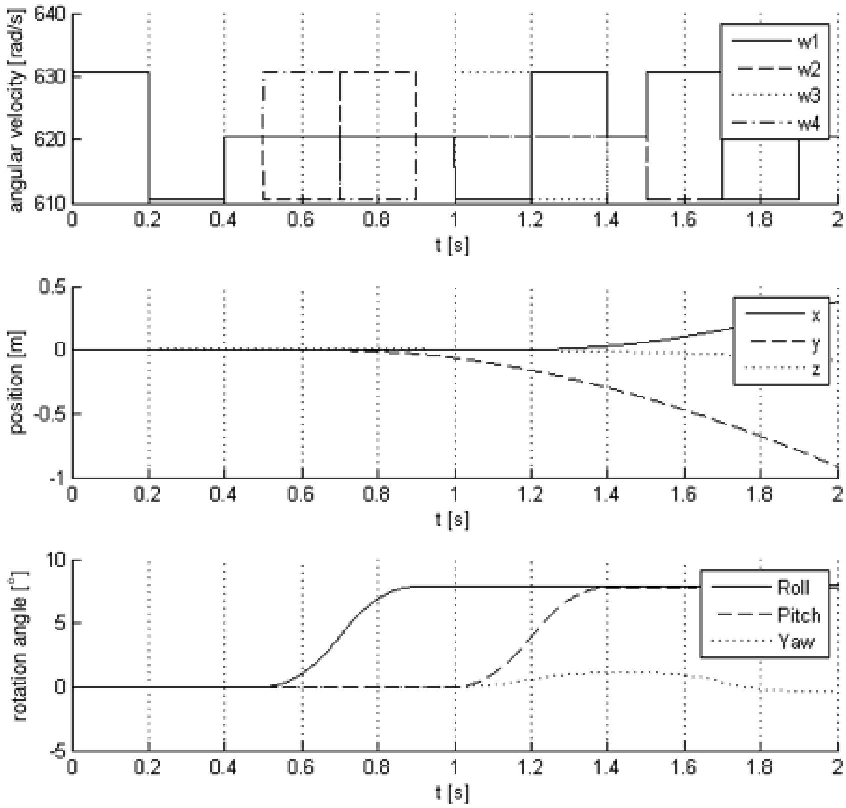

Figure 2: Time responses of the quadcopter angular velocity, position and rotation angle

4 Virtual model

During the virtual model design it was at first necessary to consider the structure of functional objects. A child changes own features always with respect to its parent, which was important in the model design. Quadcopter's most important variable is its location. This is why this feature is being animated by the functional object called "core". Rolling one object in more than one direction is complicated, and therefore three functional objects called "rot_x", "rot_y", "rot_z" were created. These objects served to animate the quadcopter rotation around each axis. To clearly see the changes of position and displacement, some objects were created showing the world coordinate system called "world_coord_zero", "w_axis_x", "w_axis_y", "w_axis_z". Other object has been created as well with names "b_axis_x", "b_axis_y", "b_axis_z". These objects were tightly linked to the quadcopter body as show the Figure 1 (with formulas). The next step was the change of quadcopter position and its displacement around the individual axes. For the complexity, object has been created with names "axis_x", "axis_y", "axis_z". Although objects followed the quadcopter in terms of its position, but they will always be parallel to the objects that form the axis of the world coordinate system. The actual quadcopter body is considered as one coherent object named "quad_body" with the exception of objects "r_1", "r_2", "r_3", "r_4" to be used in the rotors animation. Rotors has been the last created ones using the objects " rot_1 "," rot_2 "," rot_3 "," rot_4 ".

As was previously mentioned, the highest parent in this part of VRML scene is functional object "core". He will have as children functional objects "rot_x", "rot_y" and "rot_z", which will be in the order they were written, in relation parent - children. The body created as a set of different geometric objects will be the direct children of the functional object "rot_z". The quadcopter body still has as its children four functional objects that will animate the rotation of each rotor. To this functional objects the visual representation of the rotors has been created.

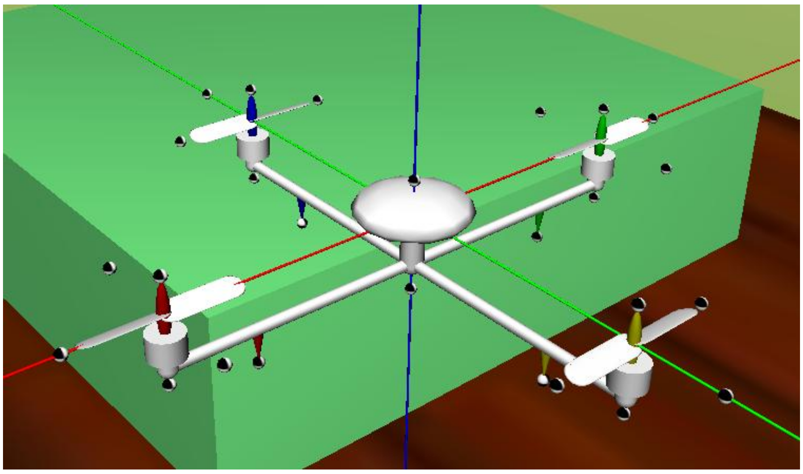

Figure 3: Quadcopter body view

It was also necessary to create the space in which it will be possible for quadcopter to move. This space is formed by the room with dimensions 5x5x2m. The exact center of the room is considered as a point [0,0,0] and therefore the lowest point in the room may enter a value in the axis of -1 and the highest is 1. Three obstacles with different dimensions were placed in the room, as well. All objects touching the floor with pedestal. The walls and ceiling are set to 50% of transparency in order to track the quadcopter movement while we had a visual idea of limited space. To ensure quadcopter detection collision with obstacles, walls, floor or ceiling of the room, it was necessary to create the twentysix collision points and place them appropriately around the quadcopter body. These points are children of functional object "rot_z". The resulting VRML scene is pictured in the Figure 4.

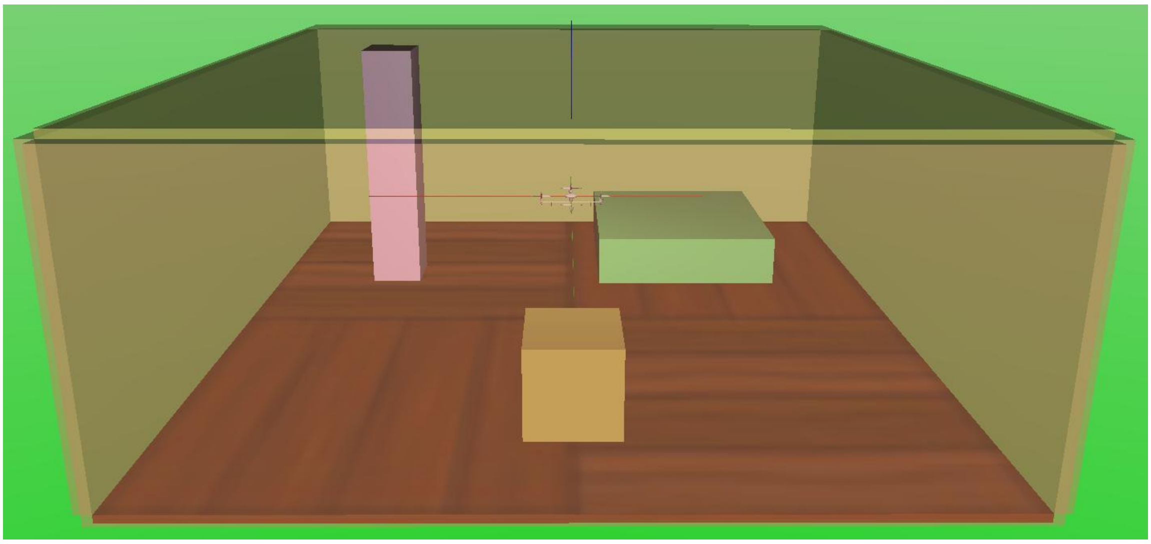

Figure 4: Final VRML scene

Since the quadcopter space flight consists of barriers, from one point of view it would not be a sufficient overview of the quadcopter flight. Thus four View-points were created. They offer a view of the overall scene of one of the four room corners. Also three View-points connected with the quadcopter body were added. The first is a view from behind the third rotor followed by the total rotation of the quadcopter, the second is a perspective view from the first and second rotors side and the last one is also a perspective view, but from the third and fourth rotors side.

5 Collision detection

To determine whether the quadcopter body does not interfere with obstacles (walls, ceiling or room floor) it was necessary to create a block of collision detection. Block Simulink 3D Animation, called VR Source gives us information about the current absolute (relative to the world coordinate system) positions of each collision points. For the correctness of the block it is necessary to load VRML world again, which is also in VR Sink. In VRML Tree is then necessary to choose which objects properties we would like to watch. For collision detection absolute position values was selected, as shown in Figure 5.

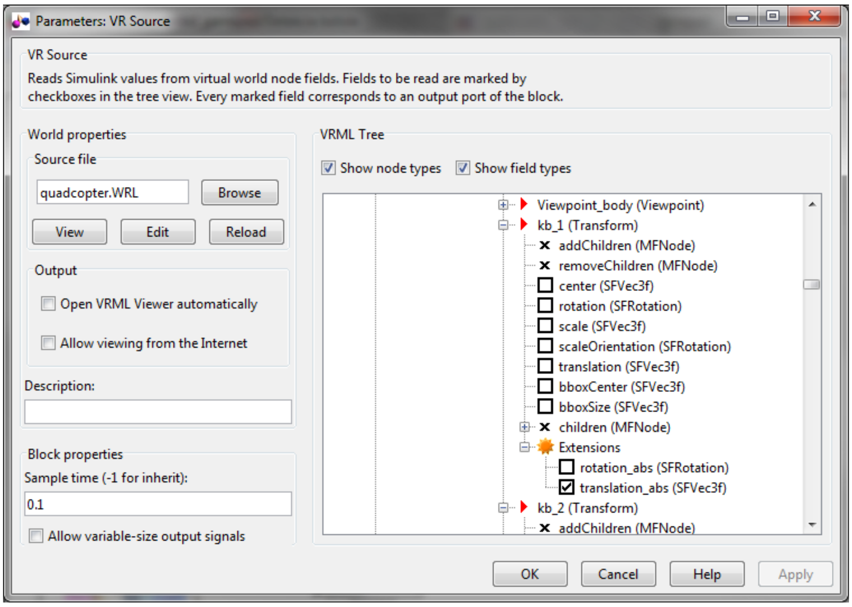

Figure 5: Absolute position of collision points monitoring option

Information about the absolute position values goes into blocks where spaces terminated with scheme objects from the Fig. 4 have been defined. These blocks are defined in MATLAB Simulink environment as Interval Test. If the condition meets the test signal from the logical operator, it stops the simulation as can be seen in the Figure 6.

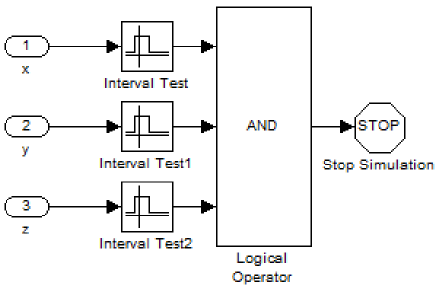

Figure 6: Block of the collision detection point and object in the space

6 Manual control

The manual control with joystick was created to the virtual model. Gamepad Genius G-12X with four-way trackpad and twelve buttons was used for control. Using the Joystick Input block, the gamepad signals generated by pressing each button can be brought into the scheme. After analysis the block's signals, exact properties were assigned to each button, as shown in the Figure 7.

Figure 7: Control items placement

To stabilize the quadcopter rotation angle it is necessary to generate a periodic signal with one sampling period. The mere button touch on the gamepad will not generate the signal, and therefore it was necessary to design a scheme. After pressed button, periodic signal with one period will be generated on the scheme output. The final scheme of this generator can be seen in the Figure 8.

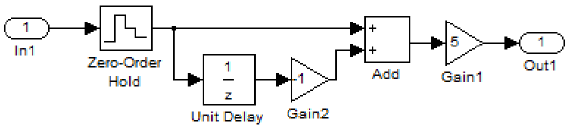

Figure 8: Block diagram of the periodic signal generator

For simplicity, the generator has been tested on the second order system (Figure 9), because the transfer function expressing the quadcopter's transmission angles to the angular velocity is also the second order system. Subsequently, a separate block for controlling the virtual model was created. When using a gamepad it is necessary to use blocks of data conversion, because the output of the block Buttons is type boolean.

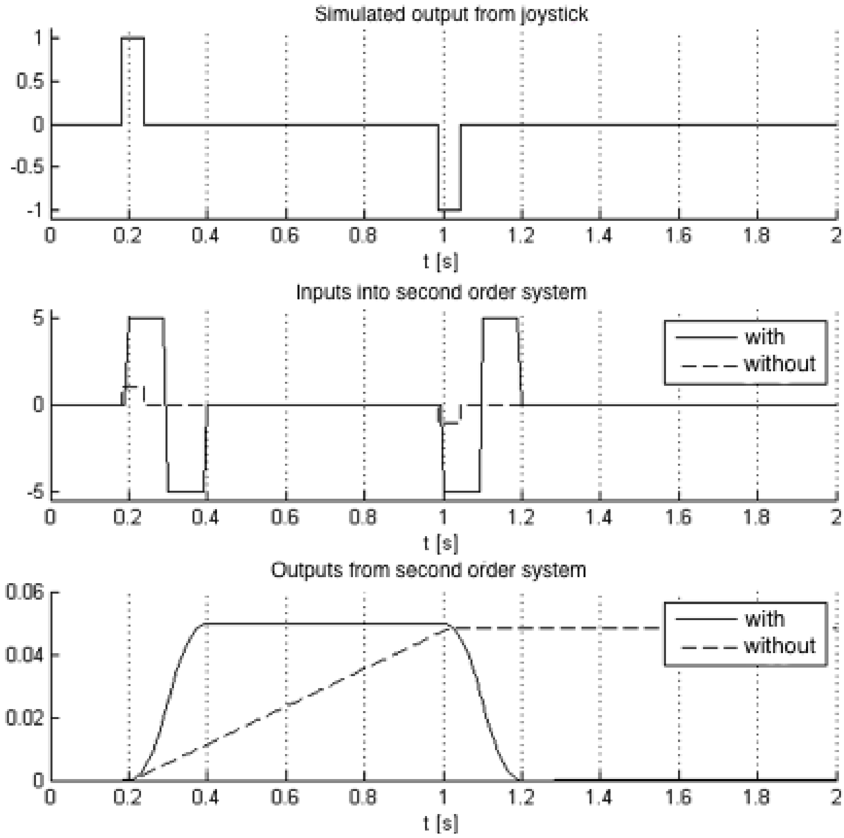

Figure 9: Time responses of the second order system output with and without generator use

7 Conclusion

This article describes the quadcopter virtual model design in the 3D environment. The simulation model in MATLAB Simulink environment was created, based on the mathematical equations from the literature [7]. Virtual model designed in the VRML language has been linked with blocks of the scheme which is why it was reached the movement in the 3D environment. Collision detection block was able to stop the simulation when facing the objects in the environment and thus gave the simulation more options. Manual control was secured by using the joystick. Quadcopter virtual model can be used in several ways. The first option is to test the control algorithms. Due to the complex structure and the links quadcopter model is suitable for their verification. Another option is to deploy the model in education. Students have the opportunity to see the device behavior in the 3D world which strengthen their interest in the design of better algorithms. The main advantage of the virtual model is that it replaces the real device that is fragile and could easily be destroyed due to some unspecified errors.

Acknowledgement

This paper was supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under the contract No. VEGA-1/1256/12.

References

[1] P. Karban. *Výpočty a simulace v programech Matlab a Simulink*. Computer Press, 2007, ISBN: 80-2511-448-1
[2] J. Žára. *VRML 97 – Laskavý průvodce virtuálními světy*. Computer Press, 2008, ISBN: 80-7226-143-6
[3] Mathworks. Simulink 3D Animation – Users Guide. http://www.mathworks.com/help/pdf_doc/sl3d/sl3d.pdf
[4] M. Foltin, M.Blaho. *MATLAB (5) - Virtual Reality Toolbox*. AT&P journal 8/2008. http://www.atpjournal.sk/buxus/docs/casopisy/atp_2008/pdf/atp-2008-08-45.pdf
[5] S. Bartoš, M. Foltin, I. Sekaj. *Dynamic plant model in virtual reality*. Technical computing Prague 2003: 11th annual conference. Praha, Česká republika
[6] M.Foltin, T.Chvostek, M. Humaj. *Vizualizácia technologického procesu prostredníctvom dde rozhrania*. Technical computing Prague 2005: 13th annual conference. Praha, Česká republika
[7] Teppo Luukonen, „Modelling and control of quadcopter“, School of Science, Espoo, August 22, 2011, http://www.sal.tkk.fi/publications/pdf-files/eluu11.pdf

---

Michal Blaho
michal.blaho@stuba.sk

Tatiana Mudráková
tatiana.mudrakova@stuba.sk