GRVC-Simulator of multirotors: design and application in classroom

A. González-Morgado, G. Heredia, and A. Ollero

GRVC - Robotics Lab Seville, University of Seville, Seville, Spain
mantonio@us.es, guiller@us.es, aollero@us.es

Abstract. The use of multirotors has grown significantly in recent years. Understanding their operation and control is essential to continue developing the field of multirotors. Because of this, it is essential to introduce multirotor control in engineering classes in a simple and effective way. However, most multirotor simulators are designed for high-level applications, making it difficult to study low-level controllers. In addition, many of them require a strong programming background. This paper presents the design and application in classroom of a multirotor simulator. The simulator developed in this work allows the easy understanding of the control of a multirotor, as it is developed in block structure using Matlab-Simulink. In addition, the block structure of the simulator allows to easily modify any part of the simulator, such as the system dynamics or the platform controller. Finally, this work presents the use of the simulator in a classroom of the University of Seville, being of great help in a practical session of introduction to multirotor control. The simulator has also helped in the development of some students' coursework.

Keywords: Multirotors · Simulators · Educational Robotics

1 Introduction

In the last years, the field of application of multirotors, aerial platforms with more than two rotors, have expanded, making possible the use of these for photography applications, monitoring crops or inspection of infrastructures [1]. Due to the increasing use of multirotors, different simulators of aerial platforms have been developed. These simulators allow testing planning algorithms, test controllers or even perform missions without the need of having the real platform.

However, the most famous multirotor simulators are designed for high-level use. Using Gazebo [2] and ROS(Robotic Operating System) [3], a simulator can be done for different types of robots, including multirotors. For example, [4] presents a quadrotor simulator based on Gazebo and ROS, which is used to prepare the competition *Drone Challenge*. In [5], a quadrotor simulator, based on ROS and Matlab, for high-level control tasks is presented. There are other simulators that are not based on Gazebo and ROS, such as AirSim [6]. However, it is still used for high-level applications, being very difficult to modify at low-level, as it happens with the multirotor control. Furthermore, it is necessary

to have a good background in programming, as most of these simulators are developed in C/C++ or Python.

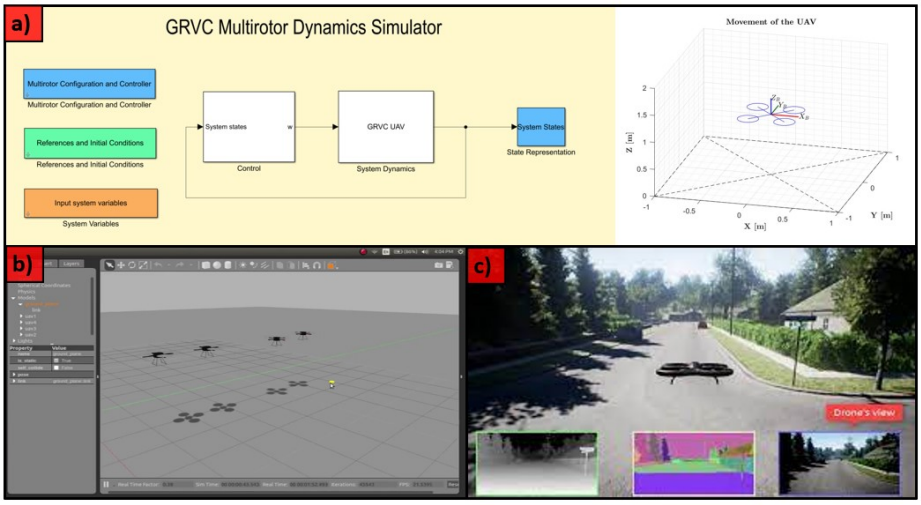


Fig. 1: Some multirotors simulators. a) GRVC-Simulator b)Simulator based on Gazebo and ROS c) AirSim simulator

These characteristics discourage the use of these simulators for educational purposes, as a first contact with multirotors. The Robotics Toolbox [7] has a Matlab-Simulink model of a quadrotor. This model could be use in classroom, since it is not necessary to have a background in programming. However, it has limited use, as it has only one quadrotor.

So, the main contribution of this paper is the design and development of a multirotor simulator in Matlab-Simulink, available in [8], which can be used to introduce multirotor control to students. As it is implemented in Matlab-Simulink, it can be easily understood by people without a strong background in programming. Furthermore, thanks to its structure in blocks, the simulator can be easily modified, allowing not only to easily test different controllers (PID cascade, geometric, backstepping, ...), but also to change the dynamics of the simulated system. The simulator has also several types of multirotors.

The paper is structured as follows. The features of the GRVC-Simulator are presented in Section 2. Section 3 introduces the structure of the GRVC-Simulator, presenting its different blocks. Section 4 shows two application cases of the GRVC-Simulator, both in the UAV (Unmanned Aerial Vehicle) course of the University of Seville. In the first one, presented in Section 4.1, the simulator is used as an aid for a voluntary exercise of the practical session. In the second, presented in 4.2, the simulator is used in different courseworks of the classroom. Finally, Section 5 summarizes the conclusions and proposes future lines of work.

2 Features of the Simulator

This section presents the features of the GRVC-Simulator. Figure 2 presents the blocks of the GRVC-Simulator, which has six different blocks. These blocks are grouped in two different zones, as Figure 2 shows, the input data zone and the control and dynamics simulator zone. The utilities of each block are presented below.

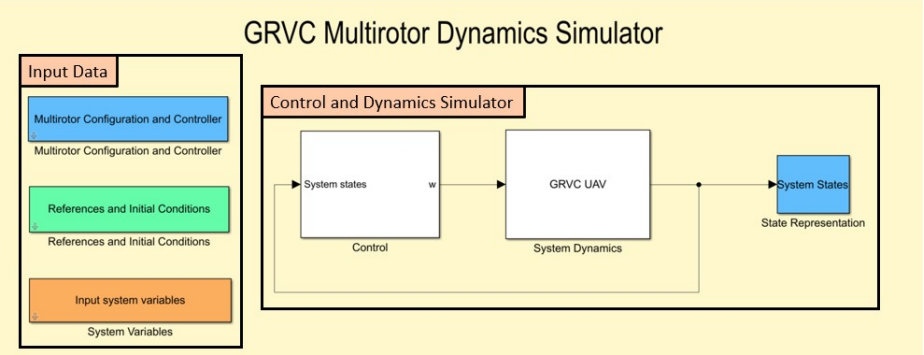


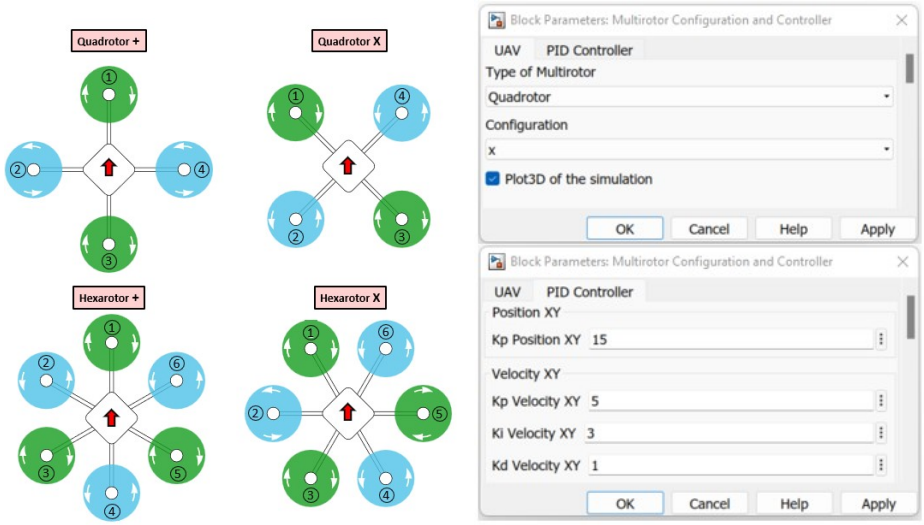
Fig. 2: GRVC-Simulator.

Multirotor Configuration and Controller. In this block, the user can select the multirotor configuration and adjust the constants of the controller. This first version of the GRVC-Simulator has two types of UAV, hexarotor or quadrotor, which can be chosen in configuration + or x. These four combinations are shown in figure 3a, which includes the numbering of the motors and their direction of rotation. In addition, figure 3b shows the interface to select the UAV and the constants of the controller. Finally, in this block the 3D visualisation of the UAV can also be activated after the simulation, as figure 3b shows.

References and Initial Conditions. In this block, the references of the multirotor are selected. The position XYZ and the yaw angle have to be selected, that are specified in form of waypoints, i.e. a value of the position or angle, and a value of time. In addition, the initial conditions have to be set in this block. Figure 4 shows the interface of the simulator to select this variables.

Input System Variables. In this block, the system variables of the multirotor can be set. These system variables are used in the System Dynamics block, in order to simulate the behaviour of the UAV. Some of these variables are from the frame of the aerial platform as the mass or the inertias I_{XX}, I_{YY} and I_{ZZ}. Others variables are from the motors and propellers, as the thrust factor, the drag factor or the maximum rotational speed of the motors. Figure 4 shows the interface to set the system variables.

Control. This block performs the control of the platform, using the constants set in Multirotor Configuration and Controller block. The controller selected for



(a) Configurations available. (b) Selection of the UAV and controller.

Fig. 3: Multirotors available and GUI for selection.

the first version of the GRVC-Simulator is the standard PID cascade, for both position controller and attitude controller, as section 3.1 presents. Using the state of the UAV, this block computes the rotational speeds that are applied to the motors.

System Dynamics. Using the rotational speed of the motors, this block compute the state of the multirotor. This block use the dynamic model of the multirotor to compute the state, as section 3.2 shows. The state is feedback to the controller, as figure 2 shows.

State Representation. Finally, this blocks have several Simulink *scopes* in order to visualize the state of the platform after the simulation.

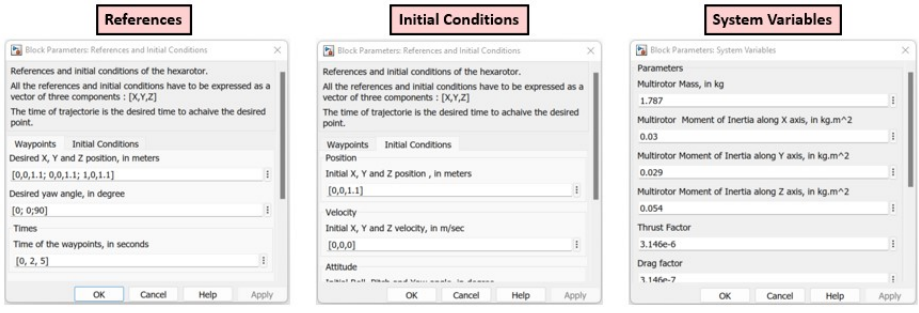


Fig. 4: References, initial conditions and system variables selection.

3 Implementation of the Simulator

As said in section 1, the GRVC-Simulator has been developed completely in Matlab-Simulink, as it is very intuitive and well known in the university environment. This section presents the implementation of the two main blocks of the GRVC-Simulator, the Control block and the System Dynamics block.

3.1 Control

The control block uses the references and the state of the multirotor to compute the rotational speed commanded to the motors. For the first version of the GRVC-Simulator, a PID cascade has been used for both position controller and attitude controller. However, the controller can be changed easily, as it has been made in a modular way, distinguishing between the position controller, the attitude controller and the mixer of the platform, as Figure 5a shows. Each block of the Figure 5a is presented below. In addition, Figure 5b shows some variables and the frames used.

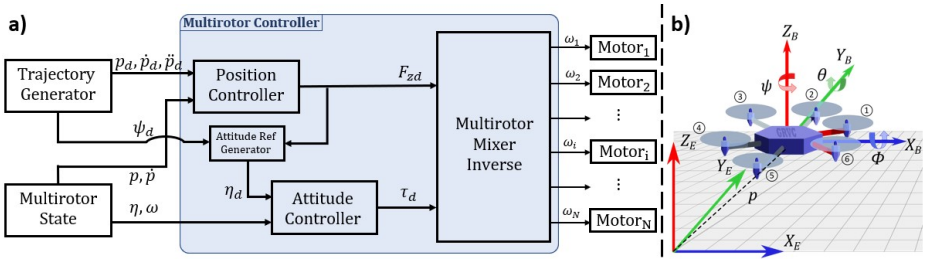


Fig. 5: Schematic of the implemented controller in the GRVC-Simulator.

Position controller. This controller is based on cascade PID theory. The control law to compute the desired force $\boldsymbol{F_d}$ is:

$$\begin{cases} \boldsymbol{v_d} = PID(\boldsymbol{p_d} \quad \boldsymbol{p}) + \dot{\boldsymbol{p}}_d \\ \boldsymbol{a_d} = PID(\boldsymbol{v_d} \quad \dot{\boldsymbol{p}}) + \ddot{\boldsymbol{p}}_d + \boldsymbol{g} \\ \boldsymbol{F_d} = m_s \boldsymbol{a_d} \end{cases} \tag{1}$$

where p is the position, $p_d = [x_d, y_d, z_d]^T$ the desired trajectory, $PID(\ )$ the PID control function, $\boldsymbol{g} = [0,\ 0,\ g]^T$ is the gravity vector and m_s is the mass of the system. The PID control function is defined as:

$$u = PID(e) = K_P e + K_I \int e dt + K_D \frac{de}{dt} \tag{2}$$

where $\boldsymbol{K_P}$, $\boldsymbol{K_D}$ and $\boldsymbol{K_I}$ are positive diagonal control gain matrices. In order to obtain the desired thrust force F_{zd}, the total force $\boldsymbol{F_d}$ is projected on the Z_B *axis* of the frame $\boldsymbol{fBg} = \boldsymbol{fX_B Y_B Z_B g}$, fixed to the body of the UAV:

$$F_{zd} = \mathbf{F}_d \mathbf{z}_B \quad \text{with} \quad \mathbf{z}_B = \mathbf{R}(\ )\mathbf{e_3} \tag{3}$$

where $\mathbf{e_3} = [0,\ 0,\ 1]^T$ and $\mathbf{R}(\ )$ is the rotation matrix from frame $\mathbf{f}B\mathbf{g}$, fixed to the UAV, to the frame $\mathbf{f}E\mathbf{g} = \mathbf{f}X_E Y_E Z_E\mathbf{g}$, fixed to the earth.

Desired orientation calculation. The desired orientation is calculated in the form of a rotation matrix $\mathbf{R}_d = [\mathbf{x}_d,\ \mathbf{y}_d,\ \mathbf{z}_d]$, whose columns are obtained as:

$$\mathbf{z}_d = \frac{\mathbf{F}_d}{\|\mathbf{F}_d\|} \quad \mathbf{y}_d = \frac{\mathbf{z}_d \quad [c_{\psi_d},\ s_{\psi_d},\ 0]^T}{\|\mathbf{z}_d \quad [c_{\psi_d},\ s_{\psi_d},\ 0]^T\|} \quad \mathbf{x}_d = \mathbf{y}_d \quad \mathbf{z}_d \tag{4}$$

where s_{ψ_d} and c_{ψ_d} are the sinus and cosine of the desired angle yaw ψ_d respectively. Using this desired matrix $\mathbf{R}_d$, the desired roll angle ϕ_d and the desired pitch angle θ_d are obtained as $\phi_d = \arctan(r_{32}/r_{33})$ and $\theta_d = \quad \arcsin(r_{31})$, where r_{ij} is the element (i, j) of the matrix $\mathbf{R}_d$.

Attitude controller. As the position controller, the attitude controller is based on the cascade PID theory. It calculate the desired torque $\quad_d$ as:

$$\begin{cases} \mathbf{!}_d = PID(\quad_d \qquad) + \dot{}_d \\ \quad_d = PID(\mathbf{!}_d \quad \mathbf{!}) + \mathbf{!}_d \\ \quad_d = \mathbf{J} \quad_d \end{cases} \tag{5}$$

where $\mathbf{J}$ is the inertia matrix of the aerial platform, $\quad$ is the measured Euler angles, $\mathbf{!}$ the angular velocity and $\quad_d$ the attitude reference, obtained from the rotation matrix $\mathbf{R}_d$.

Multirotor Mixer Inverse. With the desired thrust force F_{zd} and the desired torque $\quad_d$, the inverse of the mixer computes the rotational speed of each motor. Section 3.2 presents in detail the multirotor mixer.

3.2 System Dynamics

With the rotation speeds calculated by the controller, the System Dynamics block simulates the behaviour of the multirotor, generating the new system state. Figure 6 shows the schematic of the System Dynamics block, which is composed of different blocks.

Motor block. When the controller commands a rotational speed ω_i, the motor does not start to rotate at that speed instantaneously. It has a transient until it reaches that speed. This block represents this dynamics. A first order model has been chosen to relate the commanded speed ω_i and the real speed $\hat{\omega}_i$. This is $\hat{\omega}_i = B(s)\omega_i$, where $B(s)$ is the following transfer function:

$$B(s) = \frac{1}{\tau_M s + 1} \tag{6}$$

where τ_M is the time constant of the motor, which can be chosen as a parameter.

Multirotor Mixer. The mixer links the total force and the total torque generated in the UAV with the motor speeds. It is a matrix $H\ 2\ \mathbb{R}^{6\mathrm{x}N}$, where N is the number of motors. This matrix is obtained by adding the forces and

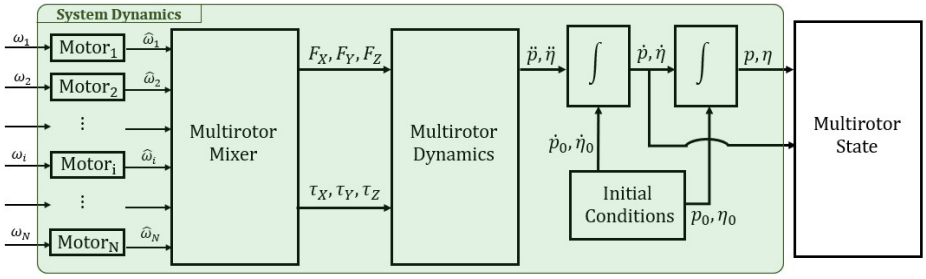


Fig. 6: Schematic of the System Dynamics block in the GRVC-Simulator.

torques relative to the centre of gravity (CoG) generated by each motor. The force T_i and torque τ_i generated by a motor-i with a rotational speed ω_i are expressed as:

$$T_i = C_T \omega_i^2 \quad \text{and} \quad \tau_i = C_D \omega_i^2 \tag{7}$$

where C_T and C_D are the thrust factor and the drag factor, respectively. For example, the mixer of the Quadrotor+ configuration is:

$$\boldsymbol{F_M} = \begin{bmatrix} F_x \\ F_y \\ F_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \boldsymbol{H_{6\times4}} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \text{ where } \boldsymbol{H_{6\times4}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ C_T & C_T & C_T & C_T \\ 0 & LC_T & 0 & LC_T \\ LC_T & 0 & LC_T & 0 \\ C_D & C_D & C_D & C_D \end{bmatrix} \tag{8}$$

where L is the distance from the CoG to the rotor position. This mixer is inverted in the control block, in order to compute the rotational speeds necessary to generate the forces commanded by the controller. If the mixer is not invertible, because it does not have a full range, the Moore-Penrose pseudo-inverse is used.

Multirotor Dynamics. This block uses the forces F_x, F_y, F_z and the torques τ_x, τ_y, τ_z to compute the second derivative of the state $\xi = [p, \eta]$, which is obtained inverting the dynamic model of the system:

$$\boldsymbol{M}\,\ddot{\xi} + \boldsymbol{C}(\xi, \dot{\xi}) + \boldsymbol{G}(\xi) = \boldsymbol{F_M} \quad ! \quad \boxed{\ddot{\xi} = \boldsymbol{M}^{-1}\left(\boldsymbol{F_M} \quad \boldsymbol{C}(\xi, \dot{\xi}) \quad \boldsymbol{G}(\xi)\right)} \tag{9}$$

where $\boldsymbol{M}$ is the inertia matrix, $\boldsymbol{G}(\xi)$ includes the gravitational forces and $\boldsymbol{C}(\xi, \dot{\xi})$ represents the centrifugal and Coriolis terms.

Integrators and Initial Conditions. Finally, the system state p, η, $\dot{p}$ and $\dot{\eta}$ is computed by integrating $\ddot{p}$ and $\ddot{\eta}$, generated by the Multirotor Dynamics block. The integrators need the initial conditions for the first step of integration. These initial conditions have been set before the simulation.

4 Application in classroom

As said in Section 1, the GRVC-Simulator was first used during the academic year 2021/22, in the UAV course of the Aerospace Engineering Degree of the University of Seville. The simulator was used in a practical session as well as in the courseworks. This section presents both use cases in classroom.

4.1 Practical session

The UAV course of the Aerospace Engineering degree has different practical sessions. One of these practical sessions consists of the control of multirotors. This practical session has five different exercises, of which the first four are mandatory. In the mandatory part, the students have to control a quadrotor that moves in the XZ-plane of the frame fEg. The GRVC simulator has been used for the last volunteer exercise, where the students have to control a quadrotor in the whole space.

Mandatory part. This part is composed of four different exercises, which deal with the control of a quadrotor in the vertical plane. All these exercises have to be done by the students, using Matlab-Simulink.

In the exercise A, the students have to implement using Matlab-Simulink the dynamic model of the quadrotor in the plane. As the UAV moves in the vertical plane, it has only 3DoF: the position x, the height z and the pitch angle θ. The simplified model of the system is provided to the students:

$$\ddot{x} = \sin\theta\frac{T}{m} \quad ; \quad \ddot{z} = g \cos\theta\frac{T}{m} \quad ; \quad \ddot{\theta} = \frac{\tau}{I_{yy}} \tag{10}$$

where m and I_{yy} represent the mass and the inertia of the quadrotor respectively, and T and τ are the total thrust and torque generated by the rotors. Once the students have created the model, they are asked to test its performance using different values of T and τ as input. Most of the students have been able to create the model correctly, as figure 7 shows.

In the exercise B, the students have to implement the attitude controller of the quadrotor, using for that the PID cascade structure. In order to have a fixed height during the design of the controller, the thrust can be fixed to:

$$T = \frac{mg}{\cos\theta} \tag{11}$$

which guarantees that $\ddot{z} = 0$, and therefore the height z remains constant. In addition, the pitch controller has to set in order to have a time constant of 0.2 seconds in closed-loop. To test the performance of the controller, the students have to test it with different types of inputs in the reference: step, ramp, sine,... As in exercise A, most students have designed and tuned the controller well.

Then, in exercise C, the position controller is designed. This controller is also based on the PID cascade structure. In order to maintain the height during the design, the thrust is still $T = mg/\cos\theta$. This controller should have a time

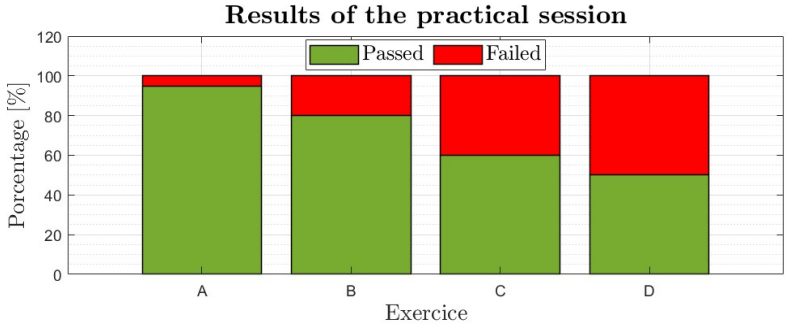


Fig. 7: Results of the practical session in the academic year 2021/22.

constant around 1.5 seconds in closed-loop. The performance of the controller should be also tested by different inputs. In this case, only the 60% of the students managed to tune the controller.

Finally, in exercise D, the height controller is designed and tested. Like the previous ones, this controller is also based in the PID cascade structure. In addition, in this last exercise of the mandatory part, the student have to check the performance of the whole system, using different references for the height z and position x.

As figure 7 shows, the pass rate is decreasing from exercise A to exercise D. The reason for this is that, with the exception of exercise A, the previous exercises must be done in order to be able to do the next one. In addition, it has been observed that, in general, students have difficulties in tuning controllers, as there are many controllers in cascade.

Volunteer part. In this exercise, students must implement the position and attitude control of a complete quadrotor, i.e. with six DoF: the position x, y, the height z and the ϕ, θ and ψ angles. They have two options: developing completely their own Matlab-Simulink model, or using the GRVC-Simulator, provided by the teachers.

Before the GRVC-simulator was developed, the students who wanted to do the voluntary section had to do the complete model. Now, they have the option to start with a model in which they have to design only the controllers, as the model is given to them with an empty control block. In this way, the students focus on the control part, as the dynamics of the system are already implemented and tested.

As figure 8 shows, the GRVC-Simulator has led to an increase in the number of students doing the voluntary part, thanks to the fact that the dynamics of the system is already implemented.

However, as figure 8 shows, there are still students who develop their own simulator. This is due to the fact that these students prefer to reuse their developed model in the compulsory exercise, increasing the degrees of freedom for this exercise.

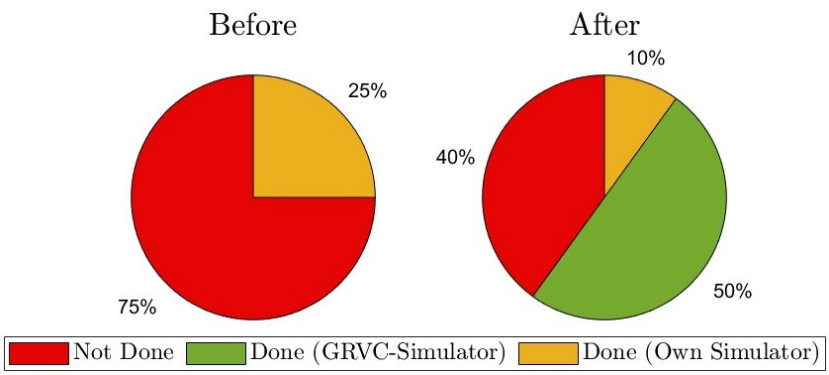


Fig. 8: Results of GRVC-Simulator in the volunteer exercise.

4.2 Courseworks

The GRVC-Simulator has also been used in some of the courseworks of the student. The coursework is a final project developed individually or in groups during the semester. This project is presented at the end of the semester. There are different types of coursework: estimation, hardware design, vision, ... However, the GRVC-Simulator can be used in projects related with control and simulation. As the simulator has a structure divided in blocks, the students can modify the block that they need for the project. The GRVC-Simulator is provided to the student who request it. This section presents projects where the GRVC-Simulator has been modified and used.

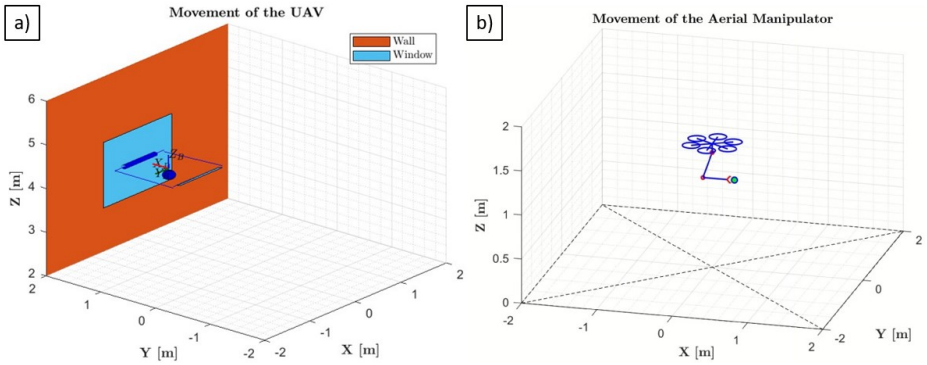


Fig. 9: Examples of use. a) Window cleaning quadrotor b) Aerial manipulator.

A window cleaning quadrotor. In this project, the students have designed a quadrotor for cleaning windows. The quadrotor has a roller to clean the glass on one side, a foam to dry the glass on the other side and a water tank at the

bottom of the quadrotor. The students have modified the dynamics in order to model the contact forces with the wall during the operation. They have also modified the 3D-representation file, in order to represent the tank, the cleaning roller and the foam to dry of the quadrotor. Figure 9a) shows the 3D-animation of the window cleaning quadrotor.

Aerial manipulator. In this project, the student has incorporated to the GRVC-Simulator a 2DoF manipulator. In order to do that, the system dynamics block has been modified, incorporating the 2DoF of the manipulator. Furthermore, the student has designed the arm controller. As a decoupled controller has been chosen, the multirotor controller has not been modified. Figure 9b) shows the 3D animation of the aerial manipulator.

Fully-Actuated hexarotor. In this case, the GRVC-Simulator has been modified in order to incorporate a fully-actuated platform. The fully-actuated platform consists on a hexarotor with tilted propellers. The student has modified the control of the platform, decoupling the position control and the attitude controller. The mixer of the system has been also modified. Figure 10a) shows the hexarotor with tilted propellers.

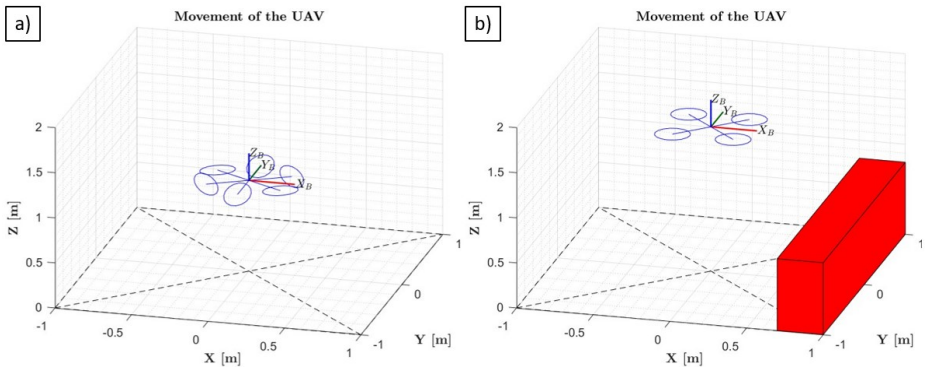


Fig. 10: Examples of use. a) Fully-actuated UAV b) Ground effect simulations.

Ground effect. In this project, the student has incorporated the modeling of the ground effect to the simulator. In order to do that, the mixer of the aerial platform is updated during the simulation, taking into account the distance to the obstacle. The obstacles can be added in the simulation using a new block in Matlab-Simulink, as figure 10b) shows. Different models can be used to model the ground effect.

5 Conclusion

This paper has presented the development and application of the GRVC-Simulator for multirotors, developed in Matlab-Simulink. The simulator has dif-

ferent types of multirotor to simulate, allowing to change the mechanics characteristics of the platform in an easy way. In addition, trajectories of reference are easily programmable using waypoints. The paper has also presented the implementation of the simulator, which is composed of differents blocks. This structure allows to make changes in specific areas in a simple way, allowing to extend the functionalities of the simulator easily. In addition, the paper has presented two different applications in classroom of the simulator. In the first one, which consists in the control of multi-rotors, the simulator has allowed an increase in the number of students interested in the voluntary exercise compared to other courses. In the second, the simulator has helped many students in their coursework, being an initial model to start with.

As future work, it is proposed the integration of others controllers for the multirotors, such as backstepping or geometric control. In this way, future practical exercises could be planned, comparing the performance of different multirotor controllers.

Acknowledgment

This work has been supported by the ARTIC project, funded by the Spanish Ministerio de Economía, Industria, y Competitividad (RTI2018-102224-B-I00), and the AERIAL-CORE (H2020-2019-871479) and the AEROTRAIN Marie Skłodowska-Curie (MSCA-ITN-2020-953454) projects, funded by the European Commission. The authors would also like to acknowledge the UAV course of the University of Seville for their feedback about the simulator.

References

1. A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, Present, and Future of Aerial Robotic Manipulators," *IEEE Transactions on Robotics*, pp. 1–20, 2021.
2. Gazebo Robotics Simulator. [Online]. Available: https://gazebosim.org/home
3. Robot Operating System (ROS). [Online]. Available: https://www.ros.org/
4. A. Bermúdez, R. Casado, G. Fernández, M. Guijarro, and P. Olivas, "Drone challenge: A platform for promoting programming and robotics skills in k-12 education," *International Journal of Advanced Robotic Systems*, vol. 16, 2019.
5. R. Spica, P. R. Giordano, M. Ryll, H. H. Bülthoff, and A. Franchi, "An open-source hardware/software architecture for quadrotor uavs," *IFAC Proceedings Volumes*, vol. 46, no. 30, pp. 198–205, 2013.
6. AirSim Simulator. [Online]. Available: https://microsoft.github.io/AirSim/
7. Robotics Toolbox for MATLAB. [Online]. Available: https://petercorke.com/toolboxes/robotics-toolbox/
8. GRVC-Simulator. [Online]. Available: https://hdvirtual.us.es/discovirt/index.php/s/dHeAq9QpNJzD93S