Simulation of the behavior of an unmanned aerial vehicle in virtual 3D scenes

To cite this article: A A Taganov and D V Shashev 2019 IOP Conf. Ser.: Mater. Sci. Eng. 516 012046

View the article online for updates and enhancements.
Simulation of the behavior of an unmanned aerial vehicle in virtual 3D scenes

A A Taganov, D V Shashev
National Research Tomsk State University, Tomsk, Russia
E-mail: asablindme@gmail.com

Abstract. The object of study is a computer-based solid-state model of an unmanned aerial vehicle. In the process of developing a simulation model, a computer-based solid-state model of the environment was created. The test flight of 3D model was made on mission points. The developed simulation model allows to accurately assess the performance of an unmanned aerial vehicle under certain flight modes.

1. Introduction
Before you start developing a simulation model, you need to select tools. It should be carefully documented; it should have a large number of libraries and extensions allowing developers to solve not only immediate tasks but also problems arising during the development of other modules of an unmanned aerial vehicle (UAV).

The simulation of the motion dynamics of a computer-based solid-state model of an unmanned aerial vehicle was carried out in the operating system for ROS robots using the Gazebo software package.

2. Robot Operating System and Gazebo
ROS (Robot Operating System) is a robot-programming framework that provides functionality for distributed work. The operating system provides the following services: hardware abstraction, low-level control of devices, the introduction of widely used functions, the transfer of messages between processes, and package management. The operating system is based on graph architecture, where data processing takes place in nodes that can receive and transmit messages between themselves. The library is focused on Unix-like systems, such as Ubuntu Linux, Fedora and Mac OS. ROS contains many open source implementations of common functionality and algorithms for robotics. These open source implementations are organized into ‘packages’. ROS distributions include many packages, while other distributions can be developed by individuals and distributed through code-sharing sites such as github [1].

Gazebo is a software package that simulates the interaction of a robot or even a population of robots with the real world. After the detailed description of a robot, it is possible to test both the work of the algorithms and the physical implementation of the robot in a virtual environment, before building it in hardware. The physics engine used in Gazebo allows taking into account such details of the robot’s interaction with the environment as friction, engine power, imitation of sensors and cameras, etc. [2]. A well-designed simulator allows you to quickly test algorithms, create robots, perform regression testing, and train your AI system using realistic scenarios. Gazebo offers the possibility to simulate robot populations in complex indoor and outdoor environments accurately and...
efficiently. The software package provides a robust physics engine, high-quality graphics, user-friendly software and graphical interfaces [3].

Using auxiliary tools, we created an open space map, which was later integrated into the simulation model using the Gazebo software package. This allows testing algorithms, performing regression testing using realistic scenarios without using real robots. In Gazebo, the required physical 3D model of a robot is described using the SDF file format, which is usually used in ROS [4].

First, we create a model of the robot in the xml-format; each functional part (base, lidar, camera, screws, etc.) is described separately, and then these parts are combined together into a hexacopter model. Such parts of a robot like cameras, motors, sensors, etc. need connection to plugins that are written in C++ or Python and describe the functionality of these parts. Plugins may be connected not only directly to the model and sensors, but to the whole world as well [5]. Figures 1, 2, 3 present models in simulated environment.

![Ground model in Gazebo](image)

**Figure 1.** Ground model in Gazebo.
3. MAVLink protocol and MAVROS
The unmanned aerial vehicle was controlled using an information protocol for interaction with drones or small MAVLink unmanned aerial vehicles. For information interaction with the device, the ROS package ‘mavros’ is used. This package implements the ability to send machine control commands through the MAVLink protocol using standard ROS data types [6]. MAVROS - MAVLink extensible communication node for ROS with a proxy for a ground control station. ROS can be used with the PX4 and the Gazebo simulator. It uses the MAVROS MAVLink node to communicate with the PX4 [7].

Lidar Velodyne VLP-16 and camera Microsoft Kinect were selected as tools of vision. Velodyne Simulator allows simulating the Velodyne VLP-16 in Gazebo. This is an open source software. The Velodyne Simulator is a workspace for models, which means that it can be built using the catkin build command in the root folder of the package [8, 9]. Working with lidar simulators requires some kinds
of auxiliary work. After building the plugin and searching for its variables in your environment, we import the lidar to Gazebo. As an example of lidar operation, let us launch the UAV environment model without and with obstacles. Figures 4 and 5 present imitation of the lidar with and without obstacles on the UAV model.

Figure 4. Imitation of the lidar without obstacles on the UAV model.

Figure 5. Imitation of the lidar with an obstacle on the UAV model.

After adding the Kinect camera model and sensor to the UAV model, it becomes possible to see through Rviz what the UAV sensors currently see. Figure 6 presents imitation of the Kinect camera on the hexacopter.
4. **Summary**
Currently, this simulation model is in the process of optimization and planning for the possible introduction of newer technologies. In parallel, modules for testing the Sense and Avoid algorithm are under development. We plan to introduce two multi-rotor UAVs to the simulation model for testing algorithms for capturing moving objects.

**Acknowledgments**
The research was carried out in Tomsk State University with the financial support of the Ministry of Education and Science of Russia, a unique project identifier: RFMEFI57817X0241.

The authors are grateful to Tatiana B. Rumyantseva from Tomsk State University for English language editing.

**References**
[1] ROS. [Electronic resource]. URL: http://www.ros.org
[2] Shikhman M., Shidlovskiy S. 2018 *IOP Conference Series: Materials Science and Engineering* 363 (1) doi: 10.1088/1757-899X/363/1/012019
[3] Gazebo. [Electronic resource]. URL: http://gazebosim.org
[4] Specification – SDF. [Electronic resource]. URL: http://sdformat.org/spec
[5] Gazebo plugins in ROS. [Electronic resource]. URL: http://gazebosim.org/tutorials?tut=ros_gzplugins
[6] PX4 Autopilot User Guide. [Electronic resource]. URL: https://docs.px4.io/en/
[7] Gafurov O. *et al.* 2012 *Telecommunications and Radio Engineering* 71(17) 1565–1574 doi: 10.1615/TelecomRadEng.v71.i17.40
[8] Kuznetsov D., Syryamkin V. 2015 *AIP Conference Proceedings* 1688 (040004) doi: 10.1063/1.4936037
[9] Shidlovskiy S. 2016 *MATEC Web of Conferences* 79 doi:10.1051/matecconf/20167901014