ROS/Gazebo-Based Simulation of Quadcopter Aircrafts

Conference Paper · September 2018

DOI: 10.1109/RTSI.2018.8548411

2 authors, including:

Adriano Fagiolini
University of Palermo
71 PUBLICATIONS **728** CITATIONS

ROS/Gazebo-based Simulation of Quadcopter Aircrafts

Claudio Sciortino, Adriano Fagiolini, *Member, IEEE*

***Abstract*—The main purpose of this work is to present a tutorial description on how to design and develop an observer, which is capable of estimating the position and the orientation of a drone commanded by a controller, whose shape and structure are unknown. Starting from Newton's and Euler's laws, a mathematical model describing the dynamics of a quadcopter has first been obtained. By linearizing this model it is possible to implement a Luenberger observer and validate it with simulations in a Linux environment, thanks to the use of the Ardupilot controller and the Gazebo simulator. Finally, starting from the results obtained from the simulation, it is possible to evaluate the error made in the estimation by observer and reconstruct the trajectory traveled by the drone during the simulation.**

***Index Terms*—Robotics, nonlinear control, ROS/Gazebo, quadcopters**

I. Introduction

RECENT progress in the field of sensors, microprocessors and aerodynamics have allowed the emergence of autonomous aircraft such as quadcopters [1]. The term quadcopter refers to an aircraft with four independently controlled rotors mounted at the ends of a rigid planar-shaped cross frame. A propeller is connected to each rotor shaft. This configuration has the main advantage of allowing vertical take-off and allowing the aircraft to be able to stay in hovering without particular problems. The quadcopter is naturally unstable, this implies that in order to have a stable flight many sensors combined with a fast and robust control system are needed.
The most important costitutive parts of a quadcopter are:

- The **frame** is the bearing structure of a drone. It has to be light and robust in order to resist to impacts and stress. It is often made of plastic or carbon fiber and has an X shape (Quad-X configuration) or an + shape (Quad-+ configuration)
- The brushless **motors** turn the propellers, supporting the drone and keeping it in flight
- The **ESC** (Electronic Speed Controls) change the rotation speed of the electric motor and control its rotation direction by varying the supply voltage supplied by the battery. Each engine has a dedicated ESC.
- The **propellers** can be made in plastic or carbon fiber and they has to be as light and robust as frame
- The LiPo **battery** is the primary source of energy, his capacity determines the flight autonomy and the maximum power developed by the engines.
- The **controller** acquires sensors data and, after an elaboration with a dedicated processor, controls the rotational speed of motors through the ESCs. The main sensors on a drone are IMU (*Inertial Measurement Unit*), that measure angular accellarion and speed, gyroscope, GPS and altimeter.

Unlike conventional helicopters, the quadcopter does not possess a tail rotor for direct yaw control, thus requiring that a more complex maneuver via torque generated by the engines. In general, given the presence of four motors, two of which rotate clockwise *(cw)* and two counterclockwise *(ccw)*, the total torque is zero as long as each pair of motors turns with the same speed. When this does not happen, the aircraft begins to turn in the air.

All these things make quadcopters useful in many areas. We find their use, for example, in building analysis, aerial photography, film and surveillance. However, one of the key challenges is to make such aircraft suitable for commercial activities. This implies the demand for accurate dynamic models, high performance and precise controllers, low latency status estimators, avoidance of obstacles and route planning, the possibility of landing on mobile platforms and picking up objects.

II. Quadrotor mathematical model

A. Preliminar notions

The position and orientation of the quadcopter can be controlled by varying the speed of the four engines. These last can be divided into two pairs depending on the direction of rotation:

- Motor 1 and 3, rotating in anti-clockwise direction
- Motor 2 and 4, rotating in clockwise direction

The quadrcopter structure has six degrees of freedom: three translational *(forward, backward and sideways)* and three rotational *(roll, pitch and yaw)*. With its four engines and its six degrees of freedom, the quadcopter is under-actuated, this prevents it, for example, a translational movement without first having made the rotation around one of its axes. To carry out this rotation it is necessary that the moment along one of the quadcopter's axes varies, increasing or reducing the thrust generated by one or more engines.

B. Coordinate system

In order to write the equations that characterize the motion of a quadcopter, it is necessary to indicate which are the reference coordinates in which the orientation and the position of the drone are described. In our case two reference systems are used, one fixed, $\mathcal{F}_0$, said inertial, representing the ground, and one mobile, $\mathcal{F}_b$, fixed to the drone. The inertial frame $\mathcal{F}_0$ was used to describe the linear position and the velocity of

978-1-5386-6282-3/18/$31.00 ©2018 IEEE

the center of mass of the quadrocopter, while, to describe the position and the angular velocity, it was used the mobile frame $\mathcal{F}_b$ attached to the drone.
It is also assumed that the $\mathcal{F}_b$ axes are aligned with the main quadcopter axes. In particular it results $\hat{i}_b$ aligned with the axis containing the motors 1 and 3 and $\hat{j}_b$ aligned with the axis containing the motors 2 and 4.

C. Euler angles

Euler angles are used to describe the orientation of a rigid body in space. In particular, these angles describe the position of a reference system XYZ attached to a rigid body through a series of rotations starting from a fixed reference system xyz. The two reference systems coincide in the origin.
To describe the orientation of the quadcopter in space, Euler angles were used in the ZXY configuration, also known as *roll*, *pitch* and *yaw* angles.
More precisely, in order to align the axes of $\mathcal{F}_0$ to those of $\mathcal{F}_b$, we first apply a rotation of the yaw angle (ψ) around $\hat{z}_0$, then we apply a rotation of the roll angle (ϕ) around axis $\hat{i}_1$ of the first intermediate frame, and finally apply a rotation of the pitch angle (θ) around axis $\hat{j}_2$ of the second intermediate frame which leads to the body frame $\mathcal{F}_b$.

The rotation matrix $R(\phi,\theta,\psi)$ converting body-frame coordinates to inertial frame coordinates is obtained by combining the above elementary rotations as:

$$R(\phi,\theta,\psi) \quad = \quad \mathrm{Rot}_z^T(\psi)\,\mathrm{Rot}_x^T(\phi)\,\mathrm{Rot}_y^T(\theta)$$

where

$$\mathrm{Rot}_z^T(\psi) \quad = \quad \begin{pmatrix} c_\psi & s_\psi & 0 \\ -s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$\mathrm{Rot}_x^T(\psi) \quad = \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{pmatrix},$$

$$\mathrm{Rot}_y^T(\psi) \quad = \quad \begin{pmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{pmatrix},$$

and it is thus given by

$$R = \begin{pmatrix} c_\theta c_\psi - s_\phi s_\theta s_\psi & -c_\phi s_\psi & s_\theta c_\psi + s_\phi c_\theta s_\psi \\ c_\theta s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi & s_\theta s_\psi - s_\phi c_\theta c_\psi \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{pmatrix}. \tag{1}$$

D. Dynamic model

The quadcopter's center of mass is subject to the gravity force, always directed along the negative direction of $\hat{k}_0$, and the thrusts applied by the four rotors, always aligned with the positive direction of $\hat{k}_b$.
Considering these forces, it is immediate to derive the following Newton's equation for the center of mass:

$$m \begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = m \begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix} + R(\phi,\theta,\psi) \begin{pmatrix} 0 \\ 0 \\ F_t \end{pmatrix} \tag{2}$$

where $F_t = F_1 + F_2 + F_3 + F_4$, and each F_i indicates the thrust applied by the i-th rotor and results to be proportional to the square of the motor's rotation speed:

$$F_i = k_F \Omega_i^2 \,.$$

In order to derive a dynamics for the quadcopter's orientation it is necessary to consider three aspects:

- The time derivatives of the Euler angles ($\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$) do not coincide with the components of the angular velocity vector $\omega = (p,q,r)^T$ expressed in the movind frame $\mathcal{F}_b$ but instead they are linked to the latter through the relationship:

$$\begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + \begin{pmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + \\ + \begin{pmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} = \\ = \begin{pmatrix} c_\theta & 0 & -c_\phi s_\theta \\ 0 & 1 & s_\phi \\ s_\theta & 0 & c_\phi c_\theta \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}; \tag{3}$$

- Since the calculations are referred to the mobile frame $\mathcal{F}_b$, during the vector derivation process it is necessary to add a corrective term consisting of the cross product between the angular rotation of the frame and the vector to be derived. Eulers equation of motion thus reads:

$$I\dot{\omega} + S(\omega) I \omega = M \,, \tag{4}$$

  where $M = (\tau_\phi, \tau_\theta, \tau_\psi)^T$ is the net moment vector and I is the inertia matrix of the quadcopter;

  In addition to forces every rotor i produces a moment M_i perpendicular to the plane of rotation of the blade. Rotors 1 and 3 rotate in the negative direction of $\hat{k}_b$, while rotors 2 and 4 in the positive direction. Since the moment produced on the quadcopter is opposite to the direction of rotation of the blades, M_1 and M_3 act in the positive direction of $\hat{k}_b$, while M_2 and M_4 in the negative one. Moreover, having denoted with l the distance between the rotors and the quadcopters center of mass, rotors 2 and 4 produce moments in positive and negative directions of $\hat{i}_b$, respective, while rotors 1 and 3 produce moments in negative and positive directions of $\hat{j}_b$, respective. These moments are proportional to the square of the motor's rotation speed according to:

$$F_i = k_M \Omega_i^2$$

- Having assumed that the body frame $\mathcal{F}_b$ is aligned with the three principal axes of the system ensure that the inertial matrix I is diagonal.

$$I = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix}$$

Based on all this, the overall expression of Eulers equation is the following:

$$I\begin{pmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{pmatrix}+\begin{pmatrix}0 & -r & q\\ r & 0 & -p\\ -q & p & 0\end{pmatrix} I\begin{pmatrix}p\\ q\\ r\end{pmatrix}=\begin{pmatrix}M_\phi\\ M_\theta\\ M_\psi\end{pmatrix} \quad (5)$$

with

$$\begin{pmatrix}M_\phi\\ M_\theta\\ M_\psi\end{pmatrix}=\begin{pmatrix}l(F_2-F_4)\\ l(F_3-F_1)\\ M_1-M_2+M_3-M_4\end{pmatrix}.$$

By developing all calculations we finally achieve the following formula:

$$\begin{pmatrix}I_{xx}\,\dot{p}\\ I_{yy}\,\dot{q}\\ I_{zz}\,\dot{r}\end{pmatrix}=-\begin{pmatrix}(I_{zz}-I_{yy})q\,r\\ (I_{xx}-I_{zz})p\,r\\ (I_{yy}-I_{xx})p\,q\end{pmatrix}+\begin{pmatrix}l(F_2-F_4)\\ l(F_3-F_1)\\ M_t\end{pmatrix}$$

where $M_t = M_1 - M_2 + M_3 - M_4$.
Finally, defining the *normalize thrust input* f and the *normalized moments* of roll pitch and yaw (τ_ϕ, τ_θ, and τ_ψ) as follows:

$$\begin{pmatrix}f\\ \tau_\phi\\ \tau_\theta\\ \tau_\psi\end{pmatrix}=\begin{pmatrix}k_F & k_F & k_F & k_F\\ 0 & 0 & -lk_F & -lk_F\\ -lk_F & lk_F & 0 & 0\\ -k_M & -k_M & k_M & k_M\end{pmatrix}\begin{pmatrix}\Omega_1^2\\ \Omega_2^2\\ \Omega_3^2\\ \Omega_4^2\end{pmatrix}$$

the quadcopter's dynamic model turns out to be [2]:

$$\begin{aligned}\begin{pmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{pmatrix} &= \begin{pmatrix}(s_\theta c_\psi + s_\phi c_\theta s_\psi)f\\ (s_\theta s_\psi - s_\phi c_\theta c_\psi)f\\ c_\phi c_\theta f - g\end{pmatrix}\\ \begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix} &= \begin{pmatrix}c_\theta & 0 & -c_\phi s_\theta\\ 0 & 1 & s_\phi\\ s_\theta & 0 & c_\phi c_\theta\end{pmatrix}^{-1}\begin{pmatrix}p\\ q\\ r\end{pmatrix}\\ \begin{pmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{pmatrix} &= \begin{pmatrix}\tau_\phi - \frac{I_{zz}-I_{yy}}{I_{xx}}\,q\,r\\ \tau_\theta - \frac{I_{xx}-I_{zz}}{I_{yy}}\,p\,r\\ \tau_\psi - \frac{I_{zz}-I_{yy}}{I_{zz}}\,p\,q\end{pmatrix}.\end{aligned} \quad (6)$$

E. Linear model

While the nonlinear model found above can effectively describe the dynamic behavior of a quadrotor aircraft in generic configuration and during generic flights, it is well known that during quasi-hovering a linearized model is sufficiently accurate. The importance and accuracy of using linearized models in quadrotors has also been recently reestablished by the work in [3]. In this vein, a linear dynamic system is a system whose evolution is governed by a linear equation, and which therefore satisfies the principle of superimposing effects. In particular, these systems depend linearly on the state variables $x(t)$ and on the input variables $u(t)$. The process that allows to convert a non-linear dynamic model to a linear one is called linearization and it is based on the study of the non-linear model around an equilibrium point considering only small deviations.
Defining the state space vector as follows:

$$X = (\Delta x \quad \Delta y \quad \Delta z \quad v \quad u \quad w \quad \phi \quad \theta \quad \Delta\psi \quad p \quad q \quad r)^T$$

with

$$\Delta x = x - x_d \quad , \quad \Delta y = y - y_d\,,$$
$$\Delta z = z - z_d \quad , \quad \Delta\psi = \psi - \psi_d\,,$$

the output vector:

$$Y = (x \quad y \quad z \quad \phi \quad \theta \quad \psi)^T$$

and the input vector:

$$U = (\Omega_1^2 \quad \Omega_2^2 \quad \Omega_3^2 \quad \Omega_4^2)^T$$

and considering as an equilibrium point the hovering state:

$$X_{eq} = (x_d \quad y_d \quad z_d \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \psi_d \quad 0 \quad 0 \quad 0)^T$$

it is possible linearize (6) obtaining the following linear model:

$$\begin{cases}\dot{X} &= A\,X + B\,U\\ Y &= C\,X + D\,U\end{cases} \quad (7)$$

where

$$A = \begin{pmatrix}0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&g\,s_{\psi_d}&g\,c_{\psi_d}&0&0&0&0\\ 0&0&0&0&0&0&-g\,c_{\psi_d}&g\,s_{\psi_d}&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\end{pmatrix}$$

$$B = \begin{pmatrix}0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 2\sqrt{\frac{k_F\,g}{m}}&2\sqrt{\frac{k_F\,g}{m}}&2\sqrt{\frac{k_F\,g}{m}}&2\sqrt{\frac{k_F\,g}{m}}\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&\frac{-l\sqrt{m\,g\,k_F}}{I_{xx}}&\frac{l\sqrt{m\,g\,k_F}}{I_{xx}}\\ \frac{l\sqrt{m\,g\,k_F}}{I_{yy}}&\frac{-l\sqrt{m\,g\,k_F}}{I_{yy}}&0&0\\ \frac{-k_M\sqrt{m\,g\,k_F}}{I_{zz}}&\frac{-k_M\sqrt{m\,g\,k_F}}{I_{zz}}&\frac{k_M\sqrt{m\,g\,k_F}}{I_{zz}}&\frac{k_M\sqrt{m\,g\,k_F}}{I_{zz}}\end{pmatrix}$$

$$C = \begin{pmatrix}1&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0\end{pmatrix}$$

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

III. State observer

A. Preliminar notions

Controllability and observability are two essential properties of a system, which are studied in modern control theory.
The concept of controllability denotes the ability to move a system around in its entire configuration space using only certain admissible manipulations. Therefore there is an external input capable of acting on the state of the system in such a way as to lead it from an arbitrary initial configuration to an arbitrary final configuration in a finite time interval.
The concept of observability denotes the possibility of being able to obtain the state of a dynamic system starting from the knowledge of its outputs and of the applied input. Observability and controllability are generally two characteristics related to each other; in particular, in linear dynamic model are mathematically dual.

In control theory, a state observer is a system that provides an estimate of the internal state of a given real system, by measuring its inputs and outputs.

In order to solve many control problems it is necessary to know the status of the system. In many practical cases, however, this state can not be determined by direct observation, but if the system is completely observable, it is possible to completely reconstruct it by measuring its outputs and using a state observer.

B. Luenberger Observer

In this work it is used a Luenberger Observer [4], [5].
Starting from the quadcopter's linear dynamic model:

$$\begin{cases} \dot{X} &= A\,X + B\,U \\ Y &= C\,X + D\,U \end{cases}$$

the mathematical model of this observer has form:

$$\begin{cases} \dot{\hat{X}} &= A\,\hat{X} + B\,U + L(Y - \hat{Y}) \\ \hat{Y} &= C\,\hat{X} \end{cases} \tag{8}$$

where matrix $L \in \mathbb{R}^{n \times p}$ it is said of *injection* of the outputs on the states.
The dynamics of error estimation $e = X - \hat{X}$ is given by:

$$\dot{e} = A\,e + L(Y - \hat{Y}) = (A + LC)\,e \tag{9}$$

and therefore it is influenced by the choice of L.
In the case where (A, C) is observable it is possible to place the coefficients of the characteristic polynomial $(A + LC)$ by changing the values of the matrix L and therefore make the error e asymptotically stable.
An L matrix making slow the convergence of the estimates is reflected in a long transient that strongly degrades the overall system's performance. Vice versa, to make this dynamic very fast, it is necessary to use high values in the L matrix, which involves larger overshooting in the transient response and a strong sensitivity to the possible measurement disturbances present in the outputs Y.

IV. Simulation Envirorment

In order to verify the correctness and full functionality of the designed observer, a series of simulations were carried out using a simulation environment comprising the following resources.

A. ErleCopter

ErleCopter is a Linux-based drone developed by Erle Robotics. We have chosen to use this model as it offers full support for ROS *(Robot Operating System)*, ensuring perfect interconnection with any device on which Linux is installed.
This drone is ideal for outdoor operations and has been designed to have a flight time of about 20 minutes transporting weights up to 1 kilogram.
The most important component of this drone is the controller Erle-Brain 3. This controller includes numerous sensors such as a gyroscope, a barometer and a thermometer. Beyond these sensors, a flight control unit *(FCU)* is also included, which provides flight controls, and a secondary processor used for optional functions such as interfacing with external autopilots (eg. *ArduPilot*) or process the images collected during the flight.

B. Ardupilot

Ardupilot [6] is the autopilot firmware officially supported by Erle-Brain 3 and is therefore the one used in this work to control ErleCopter during the simulation.
According to the developers it is the most advanced and complete open source autopilot software available, able to control any type of vehicle, from airplanes to submarines.
The advantage of being open source has allowed it to become the most tested autopilot as well as being updated and improved very quickly.

Some of the main features offered by this software are:

- Wide range of flight modes such as: acrobatics, stabilized, tracking, return to the launch site etc ...
- Automatic flight mode that executes a previously programmed path
- Real-time communication between the controller and the GCS *(Ground Control Station)*
- Complete data logging for post mission analysis

C. ROS

ROS *(Robot Operating System)* is the framework supported and used by ErleCopter for the communication between different devices [7].

ROS is a collection of tools, libraries and conventions that aims at simplifying the task of achieving complex and robust robot behaviors through a wide range of platforms. In particular, the main reason why we chose to use it in this project is that it provides the tools and libraries needed to write and start code through different computers.
A ROS system consists of a MASTER station that allows all the other components of the system (called Nodes) to interact with each other using a messaging model of publication/subscription type.
ROS Nodes do not have to be on the same system or even the same architecture. As we will see later, in our case, Erle-Brain publishes messages in various topics and a laptop subscribes to them and reads their content.

D. Gazebo

Gazebo is a simulator offering the possibility to accurately and efficiently simulate one or more robots in complex indoor or outdoor environments [8]. It has a robust physical engine, good graphics quality, and a very simple and fast interface.
We chosen to use this simulator because it is full compatible with ROS and Ardupilot and also thanks to the fact that the same Erle Robotics offers a 3D model of its drone, faithful to the original and ready to use with Gazebo.

E. Observer Simulink model

In order to allow the interconnection between the observer's mathematical model (8) and the simulated system in Gazebo, the following Simulink model has been realized:

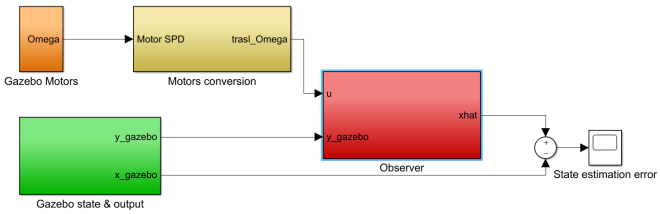

Fig. 1. Observer Simulink model

in which the various blocks have the following functions:

- **Orange and yellow blocks**
  They have the task of reading the rotor speeds of the simulated quadcopter in Gazebo and converting them into a suitable format to be used as inputs for the observer.
- **Green block**
  It has the task of reading from Gazebo the status of the simulated quadrocopter:

  $$x\_gzb = (x \quad y \quad z \quad v \quad u \quad w \quad \phi \quad \theta \quad \psi \quad p \quad q \quad r)^T$$

  and his output:

  $$y\_gzb = (x \quad y \quad z \quad \phi \quad \theta \quad \psi)^T$$

- **Red block**
  This block is the Simulink scheme of the observer (8).

V. Simulation

The interconnection between the resources listed in the previous chapter is as follows: the applications publish the

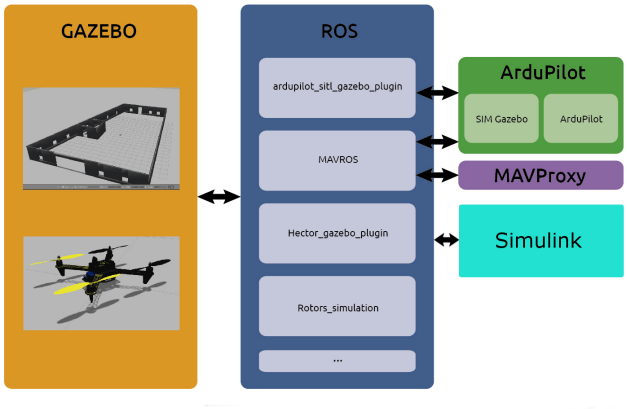

Fig. 2. Interconnection between Simulink, Ardupilot, ROS and Gazebo

data in the relative ROS topic and by registering to them they read the contents.
The ROS distribution used includes the MAVROS extension *(*Micro Air Vehicle ROS). This package introduces the drivers for the ArduPilot communication with ROS through the MAVLink communication protocol. Moreover, using this protocol, ROS is able to send commands to the simulated drone in Gazebo.
Gazebo is equipped with a mechanism that blocks the simulation until it receives the next command from Ardupilot. Once this command is received, the simulation goes ahead of $2.5\,ms$ and sends the new measurements made by the sensors to Ardupilot.
Simulink, being connected to ROS, receives the same data received from Ardupilot and then manages to proceed with the state estimation operation.

A. Simulation start

In Ubuntu, two terminal instances were started, one for Ardupilot, one for ROS and Gazebo. Through the Ardupilot terminal all the structural parameters of ArduCopter have been loaded, the main ones being:

$$m = 1.12; (1.10\,\text{chassis mass} + 0.005 \times 4\,\text{rotors mass})$$
$$Ix = 0.0347563;$$
$$Iy = 0.0458929;$$
$$Iz = 0.0977;$$
$$Kf = 8.54858e^{-6}; (\text{Motor constant})$$
$$Km = 8.06428e^{-5}; (\text{Moment constant})$$
$$l = 0.141; (\text{Rotor arm length})$$

Lastly, the flight plan was uploaded, and the position and velocity of the drone was estimated throughout the entire path by the observer realized in Simulink.

B. Results Analysis

The flight plan chosen for this simulation involves a vertical takeoff up to $90\,m$, a short period in hovering and a descent, with a circular trajectory, to the launch site.

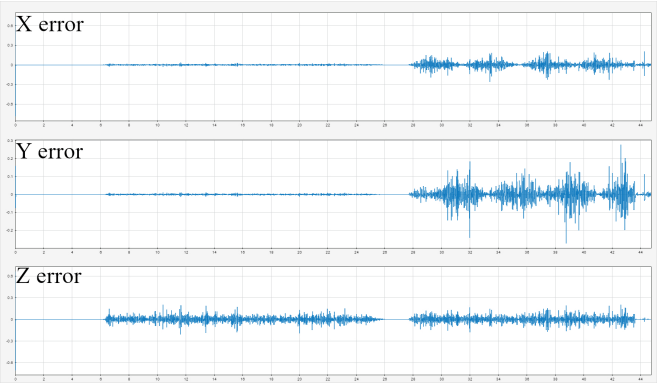

Fig. 3. Estimation of the position error

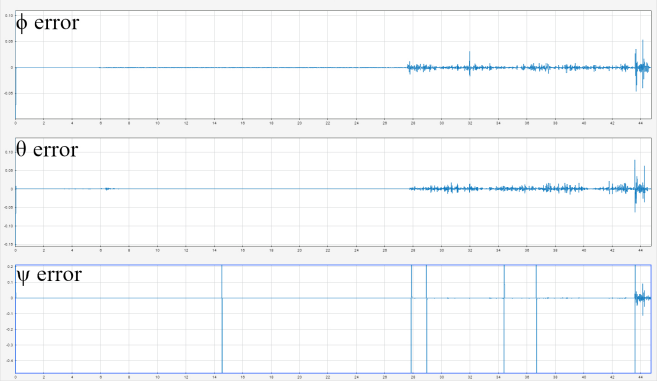

Fig. 4. Estimation of the orientation error

Analyzing the figures (3) and (4) we observe that the error committed by the observer in estimating the position and the orientation of the drone is of the order of $0.1 \sim 0.2\,m$. During the transition between a yaw value of π and one of $-\pi$ we observe the presence of spikes due to the divergence of the equation of ψ found in the linearized mathematical model.

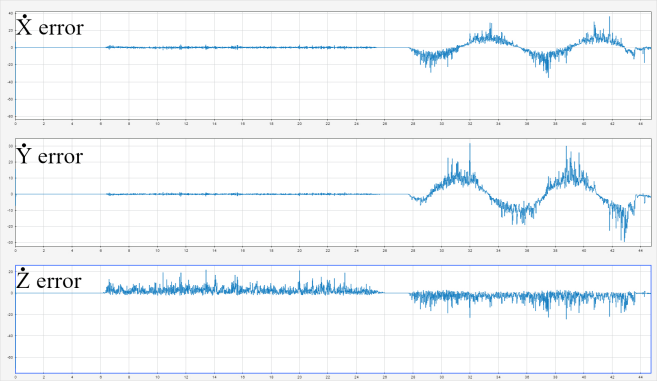

Fig. 5. Estimation of the linear speed error

Analyzing the figure (5) we observe that the error made in the estimation of the linear velocity is quite relevant. In particular, during the take-off phase up to the final height of $90\,m$, there is an error in the vertical velocity $\dot{z}$ of about $10 \sim 20\,m/s^2$ while, during the phase of landing, in the circular motion, there is an error in the velocities $\dot{x}$ and $\dot{y}$ of approximately $10 \sim 30\,m/s^2$.

Finally, from the estimated position and orientation values, it was possible to reconstruct the flight path performed by the quadcopter during the simulation:

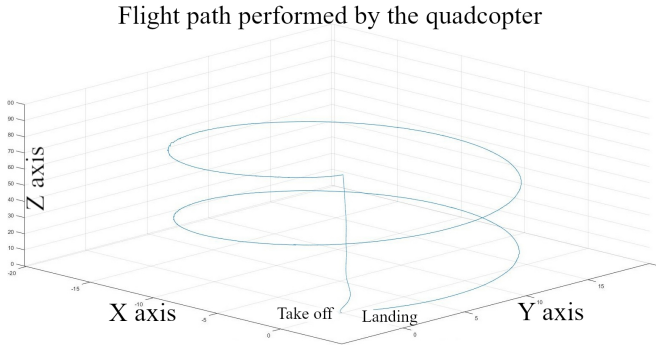

Fig. 6. Flight path performed by the quadcopter

VI. Conclusion

Starting from Newton and Euler's equations, considering the forces and moments that come into play during the rotation of the propellers of a quadcopter, we have managed to derive a dynamic model that allows to better describe the motion of a quadcopter. Through a linearization process we have obtained, starting from the dynamic model (6), a linear dynamic model used to implement the Luenberger observer (8). In order to test the effectiveness of the observer, various simulations were performed using the resources described in the section IV. Finally, analyzing the results obtained from the simulations, we have seen that the designed observer is able to correctly estimate the position and the orientation of the drone, although it presents an error of about $10 \sim 30\,m/s$ in the estimation of the linear velocities . In a future work, therefore, we will try to improve the speed estimation and to make the observer more robust to the possible external disturbances that the quadcopter will suffer.

References

[1] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav test bed," *IEEE Robotics & Automation Magazine*, 2010.
[2] A. Fagiolini, "Lectures notes on "autonomous aircrafts"."
[3] M. Bergamasco and M. Lovera, "Identification of linear models for the dynamics of a hovering quadrotor," *IEEE transactions on control systems technology*, 2014.
[4] A. Bemporad, *Lectures note on "State estimation and linear observers"*.
[5] A. Bicchi, *Appunti di "Fondamenti di Automatica"*.
[6] Ardupilot. [Online]. Available: http://ardupilot.org/
[7] Robot operating system (ros). [Online]. Available: http://www.ros.org/
[8] Gazebo. [Online]. Available: http://gazebosim.org/