Simulation and path planning for quadcopter obstacle avoidance in indoor environments using the ROS framework

Yadira Quiñonez[1] , Fernando Barrera[2], Ian Bugueño[2], Juan Bekios-Calfa[2],

[1] Universidad Autónoma de Sinaloa, Facultad de Informática Mazatlán,
Av. Universidad y Leonismo Internacional S/N, México
[2] Universidad Católica del Norte, Departamento de Ingeniería de Sistemas y Computación,
Avda. Angamos 0610, Antofagsta, Chile

yadiraqui@uas.edu.mx, fernando.barrera.pincheira@gmail.com, ibv001@alumnos.ucn.cl,
juan.bekios@ucn.cl

Abstract. This work focuses on the analysis of different algorithms dedicated to the planning of trajectories in a quadcopter. The times and distances of the paths from one point to another have been evaluated autonomously, and the evaluation of the reactive algorithms Bug1, Bug2 and DistBug have been considered to carry out the planning of the quadcopter paths. From the experiments and metrics defined, the efficiency and robustness of the algorithms have been determined. To carry out the implementation of the experiments, we have chosen and evaluated an environment based on the Robot Operating System (ROS) and Gazebo development platform.

Keywords: Bug Algorithm, Unmanned Aerial Vehicle, Robotic Software Framework, ROS.

1 Introduction

In recent years, Unmanned Aerial Vehicle (UAV) has been object of study for different researchers in the scientific community in order to develop tools that facilitate the tasks of repetitive, complex or dangerous work. For example, autonomous exploration in urban environments [1], exploration and generation of three-dimensional maps by autonomous devices of underground mines [2], support to rescue teams to avoid losses in case of landslides or when searching in inaccessible places [3], assistance in decision-making when high-risk situations occur [4, 5], among others.

UAV is a vehicle capable of carrying out a mission without needing of a crew, although it does not necessarily exclude an operator on the ground. It is possible to classify UAVs in different ways, one of the most useful is based on their take-off, dividing vertical take-offs, also known as VTOL (Vertical Take-Off and Landing), which is the fastest growing sector [6], as non-vertical take-offs, known as CTOL (Conventional Take-Off and Landing).

© Springer International Publishing AG 2018
J. Mejia et al. (eds.), *Trends and Applications in Software Engineering*,
Advances in Intelligent Systems and Computing 688,
https://doi.org/10.1007/978-3-319-69341-5_27

In this sense, calculating the appropriate movements for a machine to reach a specific point in space is a great challenge, and involves the execution of complex algorithms to achieve this end, the study of the trajectory planning has been widely developed and an endless number of methods are known to give a suitable solution in time and computational cost, which motivates and justifies to perform a comparative analysis specifically for an aerial mobile robot, in order to demonstrate the viability and performance of the algorithms. Some works focus on the bug algorithm using visual topological maps in unmanned ground vehicle and unmanned aerial vehicle [7, 8].

This work focuses on the analysis of different algorithms dedicated to the planning of trajectories in a quadcopter, where the times and distances traveled from one point to another are evaluated autonomously avoiding obstacles to define the effectiveness and efficiency of the algorithms. The analysis has been done through tests in a simulated environment, which seeks to emulate the real behavior of robots in various environments. The simulator also allows not depending on parts or sensors that may imply a cost in time or money for its maintenance. The algorithms have been implemented in the ROS development platform, an open-source robotic operating system. ROS is oriented to the specific needs of robotic platforms using reusable and modular components.

2 Robotic Software Framework

One of the basic principles of Robotic Software Framework (RSF) is to run many programs or processes in parallel that must be able to exchange data synchronously or asynchronously. For example, a RSF needs to consult sensors of the robot at an established frequency (ultrasounds, temperature sensor, gyroscope, accelerometer, cameras, among others), to recover this data, to debug them, and to direct them to the processes that are dedicated to its processing (Speech processing, computer vision, SLAM). Finally, to act on the motors or other electromechanical devices. All this process is performed continuously and in parallel. In addition, the robotic operating system needs to contain an administrator to ensure efficient access to the robot resources. Currently in the RSF there is no standard, although one of the most used for prototyping in the robotic field is ROS.

2.1 Robot Operating System

ROS is an operating system for robots created by the research laboratory *Willow Garage* in collaboration with Stanford University, is an open-source initiative for the development of robotics [9]. ROS has different tools used during programming, simulation or execution of the robot tasks, such as: Stage, Gazebo, Rviz, TF Library, OpenCV, PointCloud Library, among others.