Simulation Based Execution of UAV Missions Sent Through Web Services

Siddhartha Gupta[1], Umut Durak[2]

[1]Clausthal University of Technology, Institute of Informatics, 38678 Clausthal-Zellerfeld, Germany
[2]German Aerospace Center (DLR), Institute of Flight Systems, 38108 Braunschweig, Germany

Abstract. Drone architectures that support evolution of complex tasks as well as interoperability with other systems are vital for its current application domain. Another importantant aspect is to do a simulation-based verification of the architecture. This work outlines a system that combines an autonomous architecture, interoperability though web services and a simulation environment to verify the execution of the tasks through web services. This system was tested with the simulation on a PC and the architecture on the Raspberry Pi. The Raspberry Pi can accept nission through web services, process and execute them on the simulation enveironment remotely.

Introduction

An autonomous Unmanned Air Vehicle (UAV) has the capabilities of accepting a mission and making relevant decisions to achieve the required tasks along the way. High levels of autonomy are needed, especially when the drone is flying beyond the visual line of sight scenario [1]. A suitable feature for UAV's is the ability to work in tandem with other systems/ drones. This interoperability helps in the planning and execution of complex missions. Methodologies such as Service Oriented Architecture (SOA) especially web services, are a popular choice for exposing the functionality as a service while hiding the underlying details of the system [2]. UAV's incorporating an architecture that supports Autonomy, by supporting evolving complexity of mission tasks, and interoperability together are not explored much in the current literature. These become necessary with the advent of many upcoming applications of Drones in the military and civilian domain.

Simulations play a major role in the engineering process as they allow engineers to test designs and prototypes without spending excessive temporal and monetary resources on construction and manufacturing [3]. A computer simulation is relatively simple and convenient to deploy. The cost of any possible failiures is minimal, which encourages the developer to be creative and experiment with new features. The development time also reduces as different environments for validating the drones can be easily created in simulation [4]. It becomes quite important when the software architectures that scale in complexity on many levels. Architectures exploring topics such as Autonomy and Interoperability needs a simulation environment to coexist at development.

A useful methodology for testing and verification using simulations is Software in the Loop (SiL) testing which replaces the real sensors and actuators with simulation models and using the other systems in hardware. Simulating the drone model and its sensors makes it easier to with the architectural changes with simultaneous visualization of the results in simulation.

A popular simulator for robot-based applications is Gazebo [5]. It is a high-fidelity 3D simulator used to simulate robotic models as well as the surrounding environments. It is normally used in conjunction with Robot Operating System (ROS) [6] which is a popular middleware to develop features for autonomous UAVs. Many of the autonomy features for Drones are already available as packages in ROS. ROS and Gazebo work seamlessly with each other and uses the same communication infrastructure. It's relatively simple to create models and build an interface with ROS and Gazebo.

Architecture

This work combines three important aspects of Drone development and testing. The first one is a mechanism to interact with other systems/ drones using RESTful services. The second one is supporting an autonomous architecture that can scale with increasingly complex missions and integrating a simulation Environment to validate the new features.

The main aim here is to have a base to develop these three features independently while coexisting at the same time. The high-level architecture of our system is

ARGESIM Report 59 (ISBN 978-3-901608-93-3), p 239-240, DOI: 10.11128/arep.59.a59034

shown in Figure 1. Currently, the system can accept missions in the form of JSON scripts from another system, break its complexity down through a three-layer architecture to series of individual tasks, execute those tasks using ROS and validate the tasks in a simulated Gazebo Environment using a ROS/Gazebo communication infrastructure.

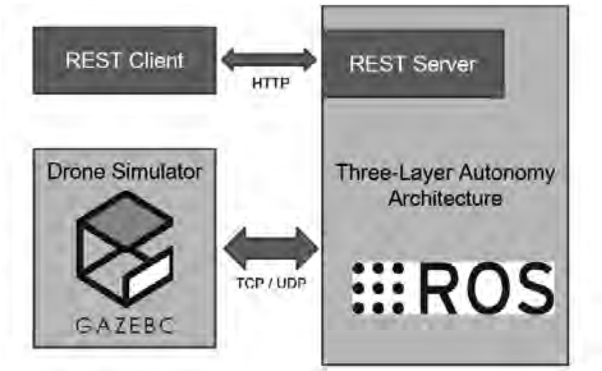

Figure 1 High-Level Architecture

We tested the system using a PC as the REST client to send a mission consisting of popular perception and motion tasks for the Drone. An onboard mission computer in the form of Raspberry Pi was used to accept the mission and another workstation was used to run the simulation.

References

[1] Viguria, A., "Autonomy Architectures," *Encyclopedia of Aerospace Engineering*, 2016, p. 1–14. doi:10.1002/9780470686652.eae1119.
[2] Mahmoud, S., Mohamed, N., and Al-Jaroodi, J., "Integrating UAVs into the Cloud Using the Concept of the Web of Things," *Journal of Robotics*, Vol. 2015, 2015, p. 1–10. doi:10.1155/2015/631420.
[3] [1] Morris, J., Zemerick, S., Grubb, M., Lucas, J., Jaridi, M., Gross, J. N., Ohi, N., Christian, J. A., Vassiliadis, D., Kadiyala, A., et al., "Simulation-to-flight (stf-1): A mission to enable cubesat software-based validation and verification," 2016.
[4] Ahamed, M. F. S., Tewolde, G., and Kwon, J., "Software-in-the-loop modeling and simulation framework for autonomous vehicles," *2018 IEEE International Conference on Electro/Information Technology (EIT)*, IEEE, 2018, pp. 0305–0310.
[5] http://gazebosim.org/
[6] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J.,& Ng, A. Y. (2009, May). ROS: an open-source Robot Operating System. In *ICRA workshop on open source software* (Vol. 3, No. 3.2, p. 5).