Demo: Developing a Fully Autonomous DJI Payload

Federico Trombetti
Sapienza, University of Rome
Rome, Italy
trombetti@di.uniroma1.it

Riccardo Tittarelli
Sapienza, University of Rome
Rome, Italy
tittarelli.1883338@studenti.uniroma1.it

Elena Valsecchi
Sapienza, University of Rome
Rome, Italy
valsecchi.1889896@studenti.uniroma1.it

Francesco Palandra
Sapienza, University of Rome
Rome, Italy
palandra.1849712@studenti.uniroma1.it

Gaia Maselli
Sapienza, Univeristy of Rome
Rome, Italy
maselli@di.uniroma1.it

ABSTRACT

In this demo, we showcase the collaborative effort that was put into bringing an offloading protocol for UAVs, Stop & Offload [1], on real-world hardware. The process includes multiple stages, from hardware configuration and testing in a simulated environment, to the final deployment in the field. The protocol enhances patrolling missions by improving coordination and data offloading among drones. We address the challenges of translating this protocol into a real hardware implementation using DJI Drones, and provide a detailed walkthrough of the process, from simulation to deployment. Our solution utilizes the DJI PSDK [2] libraries, ROS2 [5], and the Gazebo [4] simulator to ensure a secure and strong implementation.

CCS CONCEPTS

• **Networks** → **Mobile ad hoc networks**; *Network design and planning algorithms*; • **Hardware** → *Sensor applications and deployments.*

KEYWORDS

Drones, UAVs, Patrolling, Networks, Sensors, ROS

ACM Reference Format:
Federico Trombetti, Riccardo Tittarelli, Elena Valsecchi, Francesco Palandra, and Gaia Maselli. 2024. Demo: Developing a Fully Autonomous DJI Payload. In *The Twenty-fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MOBIHOC '24), October 14–17, 2024, Athens, Greece.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3641512.3690628

1 SYSTEM DESCRIPTION

We devote the next three sections to describe all the steps that will be shown in the demo. The full work is divided in three parts: i) hardware configuration and set-up, ii) running simulated environments of multiple stages (both software and in-hardware), and iii) the final deployment.

MOBIHOC '24, October 14–17, 2024, Athens, Greece

ACM ISBN 979-8-4007-0521-2/24/10
https://doi.org/10.1145/3641512.3690628

2 HARDWARE CONFIGURATION

We start by introducing the set of hardware components which allows the demo to be fully functional.
All of these hardware components are mounted together on top of a DJI drone using a custom 3d printed case, which neatly encloses the entire DJI payload in a compact and organized box.

Figure 1: The complete payload mounted on top of the Mavic 3E, with both the E-Port Development Kit, and the Raspberry Pi Zero 2W.

- **DJI Mavic Enterprise 3E**
  The Mavic 3E is the new entry in the line-up of enterprise drones of DJI; being equipped with an E-Port, the Mavic 3E can connect to different payloads, in order perform specific tasks. Payloads can be developed from scratch and be connected to the drone using the official development kit.
- **DJI E-Port Development Kit**
  The official development kit for the Enterprise lineup. It connects to the USB-C E-Port of the drone, and splits the lanes of the USB-C protocol into multiple lines that can be used to power and connect different devices to the drone.
- **Raspberry Pi Zero 2W**
  The main brain of the payload. All the libraries and instructions run from the Raspberry board. The Raspberries on top of the drones share messages with each other through wireless communication, exchanging and offloading data when the Stop & Offload protocol demands it.
  Note that the communication range and capabilities of a Raspberry are way below to what would be needed in a

realistic scenario, as their communication range with the integrated wireless antennas is really short. Nonetheless, for the scope of a visual demonstration, they allow to show a scaled-down version of a real implementation.

3 DJI PSDK AND ROS2 INTEGRATION

Our system utilizes DJI's Payload SDK (PSDK) and the open-source ROS2 wrapper [3] to interface with DJI UAVs. The ROS2 framework abstracts the complexity of direct hardware control, providing a robust communication protocol for our solution to be developed.
The flexibility of the ROS framework enables us to easily move our implementation to different drones, and to seamlessly switch between a simulated environment and a real-world one.
We corroborate the strength of our solution by testing different aspects of it with two separate simulators.

Figure 2: In-hardware simulation of two DJI drones. Two separate instances are running on their own simulation, while the Raspberry boards are still able to share messages.

3.1 DJI Assistant 2 (In-hardware simulation)

This is the official simulator provided by DJI, which accurately replicates the real behavior of the drone and requires a direct connection to ensure hardware fidelity. This simulator is particularly useful to us, as as it closely mirrors real-world performance.
The main drawback of the DJI Assistant simulator lays in the lack of ability to simulate multiple drones in a single instance, this is the reason why our solution is additionally tested on another simulated environment.

3.2 Gazebo (Software simulation)

We employ the Gazebo simulator to create a realistic testing environment for our solution. This setup allows thorough testing of the protocol in a safe, controlled manner, mitigating risks associated with direct hardware testing.
Using Gazebo, we are able to test the Stop & Offload protocol using multiple drones, before moving to the final testing on the field.

4 PROTOTYPING AND FINAL TESTING

The prototyping phase involves implementing the protocol on a single DJI Mavic Enterprise 3E drone using PSDK libraries and ROS2. This first part is important, mainly as it allows to precisely test that the behaviour of the simulated environment closely reflects the real-world one. This step is preliminary to the final demonstration.

Figure 3: Two DJI drones with their own payload set-up, before taking off.

The final testing of the protocol is conducted using three different DJIs, with each of them simulating a patrolling mission, getting executed in a pre-defined interval.
After the patrolling task is over, the drones return in range of the base station and rebuild the tree for offloading purposes. This process is repeated for the whole duration of the demo.
The exact exchange of data between drones is not detailed, as it is beyond the scope of the protocol itself, but ad-hoc messages between drones are exchanged in order to simulate real data being offloaded.

5 DEMONSTRATION

5.1 Scenario

The demo will showcase all the configuration steps needed to implement the Stop & Offload protocol, with a showcase of all the hardware used. Simulation instances, for both in-hardware and software simulation, will be displayed live during the demo.
For the real-world demo, videos of live demonstrations with three drones will be shown, as running a live demo would require a dedicated field, with multiple security concerns that may be hard to address.

5.2 Setup and Requirements

We will need a demo booth/table with power and internet access. One additional monitor would be appreciated, in order to always have pre-recorded videos running.
We will bring all the hardware described in the previous sections with us. For security concerns, drones will be displayed on the booth/table without propellers attached.
Setup time required will be approximately 30 minutes.

6 ACKNOWLEDGMENTS

This work is partially funded by the G5828 "SeaSec: DroNets for Maritime Border and Port Security" project under the NATO's Science for Peace Programme, and by Sapienza Ateneo 2021 project.

REFERENCES

[1] Bartolini, N., Coletta, A., Giorgi, F., Maselli, G., Prata, M., and Silvestri, D. Stop & offload: Periodic data offloading in uav networks. *Computer Communications 212* (2023), 239–250.
[2] DJI. Dji payload sdk (psdk). https://github.com/dji-sdk/Payload-SDK, 2024.
[3] Life, U. Psdk ros2: Opensource psdk ros2 wrapper. https://github.com/umdlife/psdk_ros2, 2023.
[4] Open Robotics. Gazebo Simulator. https://gazebosim.org/, 2023.
[5] Open Source Robotics Foundation. Ros: Robot operating system. https://ros.org/, 2024.