Addressing Challenges in FANETs - Applied UAV Experiments with Cost-Effective Hardware

Bruno José Olivieri de Souza, Markus Endler
Departamento de Informática at PUC-Rio, Rio de Janeiro, Brazil,
{bolivieri, endler}@inf.puc-rio.br

***Abstract*—The article presents the GrADyS Framework v2, which is a multi-purpose hybrid testbed for FANETs and ground devices. The framework aims to fill the gap between pure simulations and real-world validations, which remains the challenge within these testbeds. Higher-level simulation models are blended with experiments on low-end hardware by directly deploying simulation codes onto nodes like UAVs and ground sensors. Data is captured during field tests from experiments with UAV's alongside the ground units with the use of 802.11 2.4GHz networks for communication and mobility operational stability checking the bandwidth use. UAV's performance suggests that the system's effective working range extends up to 200 meters. This research draws attention to some of the practical difficulties that lie in the transition from computer-modeled environments to real-life tests and are associated with the underlying logistics of employing such models and the context in which they are deployed. This design of the framework assists evolution, eases the construction of more intricate modeling, and enhances the simulation models, which would make it possible to unravel complex FANET systems in real life.**

***Index Terms*—UAV, Testbed, FANET, real-world experiments.**

I. Introduction

A considerable portion of academic research in computer science software relies on testing and analysis for verification, particularly when formal proofs of correctness are not feasible. These tests often utilize standard datasets or, more commonly, experiments conducted in simulated environments specifically designed by researchers. Simulation plays a critical role in fields such as computer networks, distributed systems, and optimization, as it allows researchers to progress more efficiently by eliminating the need for costly equipment and mitigating the risks of damage that could render equipment unusable for further experiments.

However, while simulations facilitate research and enable scenarios that might otherwise be inaccessible, they inherently simplify environmental variables. This simplification becomes more pronounced when researchers use custom-built implementations instead of well-established frameworks, such as NS3 or OMNET++/INET, particularly in the context of network-related proposals.

Real validations—or "realistic experiments[1]"—are pivotal in assessing the impact of environmental factors (such as sunlight, temperature, humidity), scenarios, devices, and equipment on results. Such experiments enabled fully coupling with reality—consist of the high accuracy learning parallels. As far as active research for cyber-physical systems is concerned, these verify simulations by exposing practical discrepancies that are essential for tailoring the models, software, and hardware.

This work in this paper contributes by sharing new experiences in the development, utilization, and maintenance of a test-bed designed for validating proposals and studies addressing issues related to networks and distributed systems within the context of Flying Ad-Hoc Networks (FANETs) and devices on the Ground. The new test-bed, referred to as the GrADyS Framework v2, comprises simulators, autonomous UAVs (drones), and sensor networks that are effectively deployed in the field, along with other equipment such as autonomous ground vehicles (UGVs).

Additionally, field measurement results are presented and their respective implications are discussed. The GrADyS project aims to conduct both simulated assessments and field validations to foster discussions on drone swarm coordination[1][2].

II. GrADyS Framework v2

The GrADyS Framework v2 integrates multiple components, with its primary strength residing in its capacity to validate simulations involving drones and rovers and, in some instances, IoT/WSN devices in real-world environments. In addition to inherent improvements to the implementation of the systems involved, the two main enhancements to the original framework[1] were (1) the introduction of an application layer that enables researchers to perform rapid prototyping in Python [3], and (2) the capability to use the same simulation code to execute distributed algorithms directly on the nodes, such as drones.

The framework is structured around four key pillars: (1) Ground simulations of WSN and routing algorithms; (2)

[1]https://youtu.be/rz02rnxYYQY?si=hISwZ3T3D89Gy1dZ

Air simulation of protocols and distributed algorithms for coordination of mobile nodes with ad hoc communication; (3) The effective implementation of a WSN with ground sensors; and (4) Coordination of UAVs in real flights;

The simulation of protocols and distributed algorithms for mobile node coordination is facilitated through the use of Python-based coordination code designed for simplified execution to enable rapid prototyping in both simulations and vehicles. Concurrently, this work can leverage the OMNET++/INET suite to simulate communications with greater fidelity, as well as to coordinate the movement dynamics of aerial vehicles within a Software-In-The-Loop (SITL) simulator for real flight controllers based on Ardupilot stack. Specific results related to this line of research can be explored in the GrADyS-SIM project[4][3], which is based on OMNET++/INET, and MAVSIMNET[2], a tool that integrates the Mavlink protocol with OMNET++/INET.

The coordination of UAVs in real flight scenarios is carried out using low-cost, basic quadcopters, which are individually assembled by students using off-the-shelf components. These UAVs are equipped with single-board computers (SBCs), typically Raspberry Pis, allowing for the implementation of protocols and distributed coordination algorithms while supporting 802.11 and 802.15.4 based radios. Testing with actual vehicles introduces a higher degree of confidence in the validation process while also presenting new challenges in executing these tests.

In the real-world implementation, the framework is brought to life through various components, as depicted in Figure 1, which illustrates both the simulation elements and the field testing components.

The GrADyS Framework is divided into two main focuses: (1) Enabling simulations with high reproducibility, high abstraction, and high accuracy. This is achieved by making all the code available on GitHub, where robust simulators like OMNET++/INET are encapsulated within more practical simulators, allowing for rapid prototyping; (2) Providing a real-world testbed where devices and UAVs can be tested in natural environments. This approach facilitates the Verification and Validation ($V\&V$) of the simulated proposals.

III. Related Work

In swarm UAV coordination protocol research, several studies implement simulators to present and verify their findings. Notable works include AirSim[5], FlyNetSim[6], GrADyS-SIM[4],GrADyS-SIM-NG[3], ArduSim[7], and many others.

The vast majority of studies focus on simulation, and typically, the portion related to communication between nodes is less reliable and simplified. Exceptions to this statement, to the best of our knowledge, are present when there is synergy with ROS environments and in MAVSIMNET.

The implementation of reusable real-world testbeds for the validation of proposals with UAVs has been around for just over a decade, starting with the work of Vijay Kumar et al.[8] at the University of Pennsylvania. In this work, various micro UAVs communicate with a central node that controls them with a centralized Matlab model. The entire positioning system is indoor-enabled with image capture systems[9], such as VICON hardware. Following this work, other reusable testbeds with the same architecture were implemented by academia, such as at ETH by Rafaello D'Andrea et al. since 2014[10], or even with underwater vehicles, as in the work of Sidney et al.[11].

Over a decade later, new works are presented with a similar architecture, as in the case of Guerreiro et al.[12], where the notable difference is that the positioning system of UAVs is controlled by UWB (ultra-wideband), and control remains centralized, emulating independence between nodes and issues in FANETs.

In the mentioned cases, there is always a lack of outdoor tests, which are invariably more challenging to manage in terms of interferences, whether environmental, telecommunications-related or simply logistical. Another factor, perpendicular to all the mentioned works, is the processing of emulated nodes or real vehicles, which is often carried out on a central node with high processing power. In our work, distributed algorithms are effectively tested on separate nodes that communicate in a fully Ad Hoc FANET."

IV. Lessons Learned / Our Real-World Experiences

Even in more mature research groups or thematic laboratories with multiple researchers in robust university departments, research often occurs compartmentalized or even individualized. Around the same theme, several postgraduates and young researchers conduct their research and implement their simulations individually. This becomes practical in terms of verifications. However, it *may* hinder the reuse and reproducibility of research.

When we start talking about validations, this scenario encounters two main obstacles: the first is that the costs associated with transitioning from simulations to real-world testing almost always lead research teams to question how to reuse the investments made. Secondly, a single researcher is unlikely to be able to implement and execute real-world experiments with various equipment in a huge area[3]."

A. "Leaving the laboratory"

The initial step is often the most challenging. Acquiring equipment requires thorough planning and even more time to secure adequate funding, which is frequently insufficient.

[2]https://thlamz.github.io/MAVSIMNET/

[3]Video at https://youtu.be/VCO6h4jAAQk?si=SzdE1psVXQYvac-m

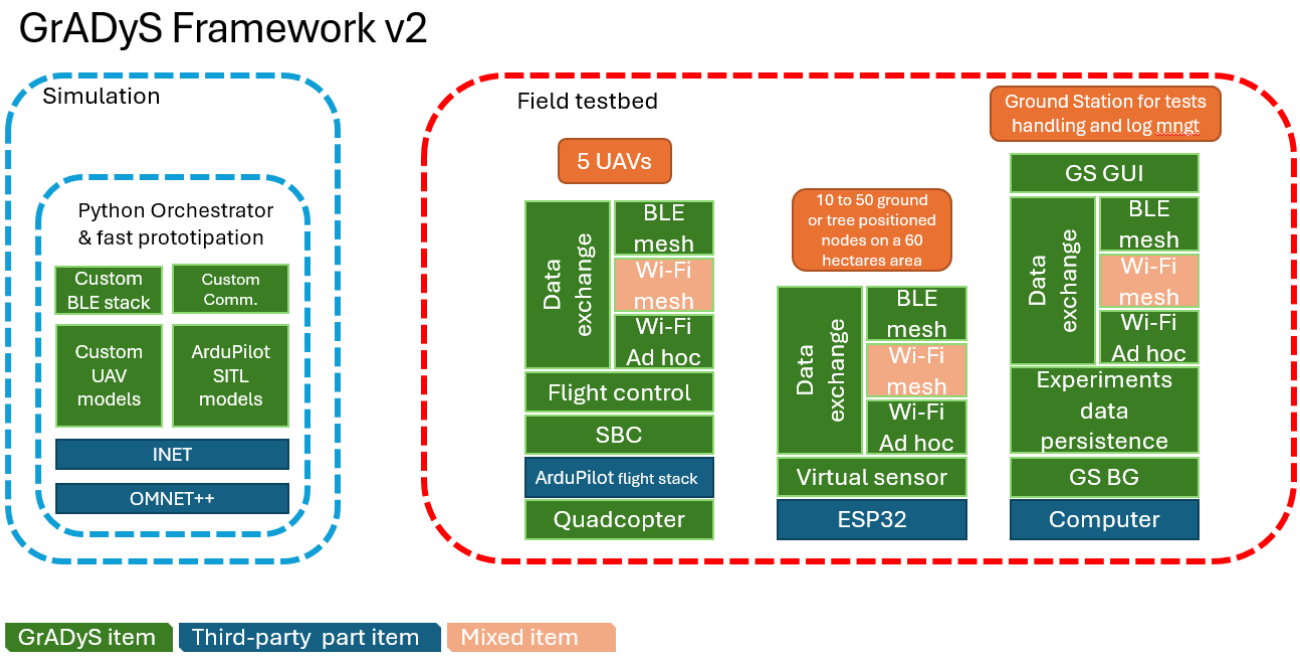


Fig. 1: The GrADyS framework illustrating its simulation components in the laboratory and its field components.

In the case of the GrADyS testbed, described in Section II, the drones are equipped with at least three radios, necessitating complex logistical planning. Beyond the drones, various other equipment such as laptops, antennas, support infrastructure, and sufficient space are required, with the latter two demanding special attention.

The Figure 2 provides a summary of the process we consider to be the most suitable for field validation tests.

i **`Definition`**: The definition of what will be evaluated in the field must be as detailed as possible, including the metrics and expected outcomes;

ii **`Planning`**: A comprehensive plan covering all aspects of the field experiment must be developed within the context of the study. Simulations are highly recommended at this stage, as they help assess whether the proposed metrics are reasonable and determine how they will be measured and collected;

iii **`Yes?`**: Simulations' outputs are reviewed and evaluated as though they were the actual experimental results;

iv **`Adjusting of plans`**: If the simulation results do not align with the `Definition`, the experiment must be replanned;

v **`Preparing`**: While planning focuses on the logical aspects of the experiment, the `Preparing` phase emphasizes the physical setup. All equipment should be thoroughly checked, and preliminary tests conducted. The use of comprehensive checklists is highly recommended at this stage;

vi **`Field Experiments`**: The field experiments themselves typically last significantly longer than any of the other phases. No activity should be underestimated in terms of duration; for instance, positioning the nodes as simulated can easily take more than an hour;

vii **`Data Collection`**: Whenever possible, all data collection should be automated. Manually collecting data from distant nodes is highly time-consuming. Additionally, data cataloging must be flawless, as unlike simulations, successive tests may become difficult to track and distinguish;

viii **`Checking data completeness`**: After data collection, it is advisable to implement a customized health check for the data to ensure that all planned nodes generated data and that it was properly collected. As the number of nodes increases, it is common for at least one node to fail or not generate logs, which may necessitate repeating the experiment;

ix **`Try to fix something`**: This is the least efficient

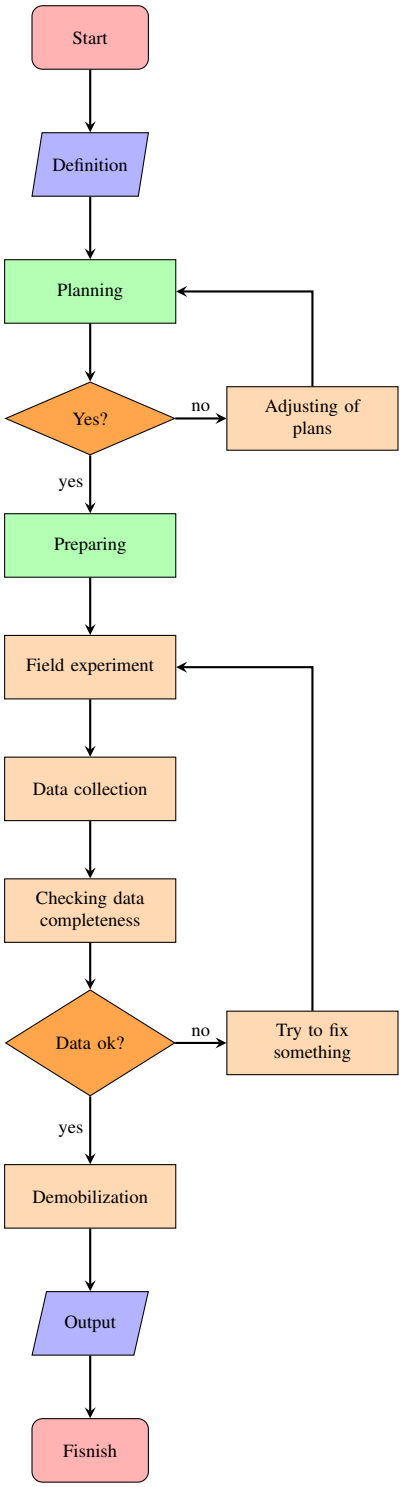


Fig. 2: Field experiment flowchart

task that can be performed and should be avoided. Priority should be given to simplifying or even reducing objectives to avoid this. Although far from ideal, it is common for detected errors in the previous phase to undergo debugging in the field, and in many real-world cases, this leads to fixes in the source code related to the experiment. While we strongly advise against this, if such fixes are made, they should be done using a version control system. In our case, a backhaul network enables internet access for nearly all nodes, allowing updates to be replicated via GitHub, alongside various log collections;

x **`Demobilization`**: Dismantling the experimental setup can take a considerable amount of time, particularly at the end of a long workday when natural light may be limited;

xi **`Output`**: Finally, the collected data can be transferred to the lab for detailed comparison with the simulation results.

On some university campuses, space may be non-existent, while on others, it might be remote and barren. For experiments involving drones and WSNs, as previously described, our testbed covers an area of 60 hectares. In our experiments, we deployed sixty ESP32-based nodes across the field; however, rearranging or replacing batteries in exactly the same locations as simulated proved extremely difficult. In practice, our most efficient tests were conducted within a 10-hectare area, which took approximately one hour to (re)configure.

The absence of basic infrastructure, such as tables and power outlets, significantly complicates outdoor testing. Conducting tests during rainy days is impractical, but even on sunny days, outdoor testing presents challenges. Sunlight renders laptop screens unusable, making accessories like umbrellas, boxes, or gazebos essential. Portable tables, long extension cords, and walkie-talkies are indispensable for outdoor tests. Images from these testing sessions can be accessed at https://gradys.tumblr.com/.

B. Vehicle in a FANET

By their very nature, vehicles are included in experiments to move, and drones extend this to a three-dimensional environment. Outdoors, they are highly susceptible to wind and hazardous obstacles. Preparing the vehicles is time-consuming, and at least one person is required per vehicle at time. If it takes ten minutes to prepare a single drone and there are six drones, either six people are needed, or it will take over an hour to complete the setup.

The decision between acquiring pre-built drones or assembling custom prototypes can be challenging. For instance, in

real-world FANET tests[13], drones serve as mobile nodes that, while not the primary focus of the research, are nonetheless essential. Moreover, research groups focusing on scientific advancements in FANETs may lack the expertise to assemble, maintain, pilot, or automate drones effectively.

Consumer-grade drones, such as those from DJI, are popular due to their ease of use but lack support for additional payloads and advanced customization. A thorough analysis is essential before selecting drones to ensure they meet long-term needs. While high-end DJI drones are suitable for integration with systems like ROS, DIY vehicles using OpenHardware and OpenSoftware, such as those supported by the ArduPilot stack, offer greater customization and reusability, though they require a longer development period. Managing drones involves significant logistical challenges, as errors in data collection can result in the need to repeat experiments, demanding meticulous planning and execution.

C. Batteries

Drones commonly use lithium batteries, such as LiPo or LiIon, which provide high discharge power but are costly and regulated due to transportation and storage restrictions. These batteries typically charge within an hour but offer only 10–20 minutes of flight time, requiring multiple batteries and parallel charging for extended testing. Additionally, proper management is necessary, as lithium batteries should not be stored fully charged or discharged for storage.

In our experiments, the most balanced field setup in terms of setup and deployment times is to use a battery capable of running an experiment twice and to have 2 to 3 batteries for each mobile node. More than that, it is usually unnecessary and involves work to discharge batteries.

D. Radios & Data links

After addressing FANET vehicle challenges, radios become critical. They fall into two categories: 802.11 radios for structured or large data exchanges and 802.15.4-based radios (e.g., LoRa, ZigBee) for minimal data over longer distances. 802.11 provides extensive interoperability and robust libraries, while 802.15.4 offers simplicity and range but faces fragmented hardware and limited library support. Practical experience guides the choice between these types. In our experience, 802.15.4 is more versatile and useful for products while 802.11 is versatile enough to be used and much better for the research environment. Basic use of 802.11 in the context of WiFi may be daunting, but in the context of ad hoc communication with one or more UAVs, with no real infrastructure, it becomes very useful and with much longer ranges.

E. Antennas

In addition to radios, the use of antennas requires special attention. Many devices, such as certain ESP32 models and the Raspberry Pi, come with antennas printed directly onto their boards. While these antennas perform adequately over short distances, their range becomes severely limited when mounted on drones, reducing it to just a few meters and making tests impractical. The motors of drones generate significant circuit noise, so the use of low-pass filters is essential.

Prioritizing external, high-quality antennas, separate from the equipment, is crucial. The market is saturated with antennas that are improperly cut for the intended wavelength. In our testbed, drones equipped with Raspberry Pi and their original antennas achieved a range of only 10 meters during flight. By switching to an external antenna and USB radio, we extended the range to 40 meters. Finally, modifying the original Raspberry Pi radio by removing the standard antenna and installing a custom-designed one, we reached a range of 200 meters, which is the maximum expected for the 802.11.

F. Companion Computers

For experiments involving distributed systems, the nodes must be logically independent. In the case of drone swarms executing distributed algorithms, embedded processing nodes are essential. These nodes require the radios and antennas previously discussed.

Since the drone handles only flight control processing, the experiment must run on a companion computer embedded within the drone. Two common options for this role are: (1) System on a Chip (SoC) devices like the ESP32, and (2) Single Board Computers (SBC) such as the Raspberry Pi, NVIDIA Xavier, and similar devices. Each option presents its own set of advantages and disadvantages.

i **SoC**:

- Advantages: They are very small and easy to embed, lightweight, and inexpensive. They are also more durable than SBCs and can work for a long time on batteries;
- Disadvantages: Limited processing power; it's important to note that the logical processing cores for the inserted code may be the same as those processing radio packets, leading to potential interference. Some programming paradigms are not available, such as multiprocessing. The code is usually manually inserted ('flashing' the firmware), making it impractical and challenging to customize. There is limited reuse of other open-source research.

ii **SBC**:

- Advantages: Good processing power, ample memory, multiple embedded radios (at least WiFi and BLE), highly expandable, use of operating systems, and other possibilities similar to a complete computer, such as a laptop. There is a lot of reuse from other open-source research and projects;

- Disadvantages: They are more expensive. In fact, a Raspberry Pi 4 with 4GB currently costs the same as a mini PC with an Intel N5095 processor, which has many more resources but is not easily embeddable. SBCs do not work as long on batteries as SoCs and are not always small and light enough to be embedded.

iii **`Tradeoffs`**: While system-on-chips (SoCs) offer a quick start and early successes, they are limited in resources, making it difficult to manage tasks outside the experiment's core. These challenges arise in parallel processes, such as data collection, log storage, and organization. In our experiences, even when using advanced SoCs like the Nordic or ESP32, integrating a single-board computer (SBC) between the SoC and the drone became essential for managing the experiment, collecting data, and controlling the drone. Furthermore, embedding an SBC instantly provides access to a wide range of tools available in the operating system, including Python, ROS2, databases, and messaging brokers.

G. Operational system for the Single Board Computers

Android and standard Linux, though both built on the Linux kernel, are designed for distinct purposes. Android is optimized for mobile devices, providing a more restrictive and user-friendly interface, while standard Linux distributions like Raspberry Pi OS offer greater flexibility and customization. These Linux distributions provide full access to a comprehensive operating system, making them ideal for versatile tasks on the Raspberry Pi, such as development, server hosting, and DIY projects. The lightweight nature of Linux ensures efficient performance on the Pi's limited hardware, making it better suited than Android for embedded systems and broader computing applications.

H. Logs and synchronization

A significant challenge in network and distributed systems research is the time-intensive process of debugging code, which frequently operates asynchronously or in parallel. To mitigate this, many IDEs offer features for monitoring values in parallel, while developers often rely on extensive logging.

However, real-world experiments introduce complications not encountered in simulations. System-on-chips (SoCs) and single-board computers (SBCs) typically lack an onboard Real-Time Clock (RTC) with a battery. Even when memory cards are employed for logging, synchronization remains problematic. SBCs usually rely on the Internet for clock synchronization, which may not be available in a FANET environment. Although RTCs can be manually installed, they tend to lack precision and necessitate additional customization of the drone's electronics.

One solution we implemented involved using the drones' APIs to retrieve highly accurate time data from the GPS. While seemingly straightforward, this approach proved labor-intensive.

Even with synchronized logs across all nodes, these logs are dispersed among the various drones, sensors, and nodes. Collecting them can be as time-consuming as the battery charging process. In this context, a Ground Station (GS) becomes invaluable, enabling data collection in the field to verify their integrity before analyzing the experiment post-fieldwork. Managing multiple SD cards and files, however, remains a cumbersome task.

I. Ground Stations

Ultimately, every experiment hinges on data collection. In our prior experiences, each student typically began by independently developing a set of scripts, resulting in a fragmented collection of code and analyses. This occurred when their efforts did not evolve into fully developed Ground Stations (GS) for controlling drones and experiments.

Such efforts could be more efficiently leveraged if planned with reusability in mind, across multiple research projects. The common objective in most cases is to monitor activity, gather data, and issue commands to a subset or all drones.

With this in mind, we developed and implemented GrADyS-GS[14], a web-based GS designed for ease of maintenance and customization. Written in Python and JavaScript, it facilitates simple data exchange and storage via JSON between vehicles and the GS. This system enables multiple users to observe the same experiment on different terminals and collaborate in real-time to control various aspects of the experiment.

V. Some results and lessons learned

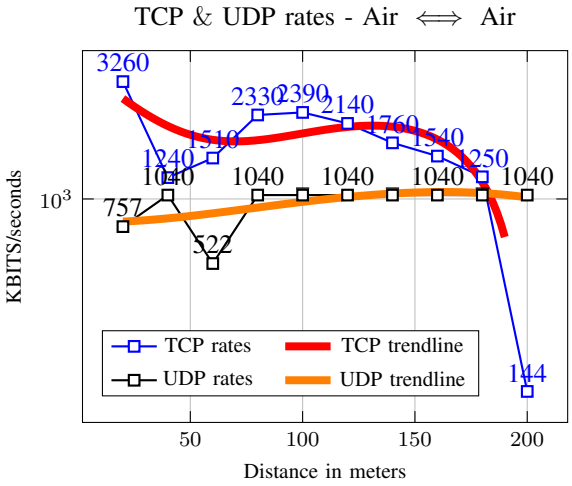
In all our experiments, we explore paradigms without centralized infrastructure for communications, with the GS being only an Ad Hoc node (802.11), but stationary. The radios used were the RPi's native radios with 2.4Ghz Wifi and half-wavelength antennas.

A. 802.11 Ad Hoc - Air-Air

In our ongoing efforts to advance FANETs for coordinating drone swarms, we transitioned to using Single Board Computers (SBCs) running Linux embedded in the drones. On these SBCs (Raspberry Pi 4), we physically modified the hardware by removing the traces connecting the radios to the printed antennas and soldering ipex connectors[4]. This modification allowed the attachment of high-quality external WiFi router antennas to the SBCs.

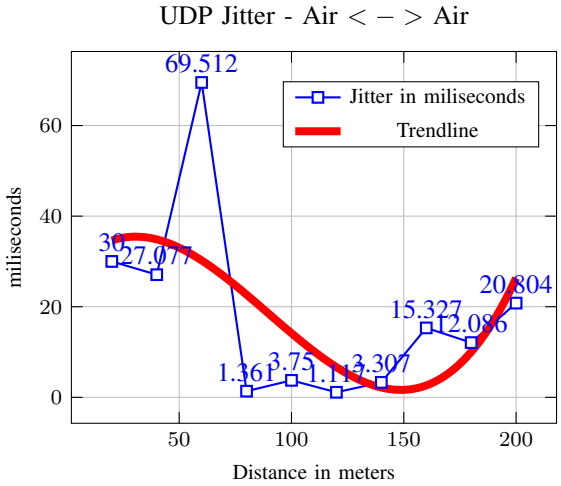
We configured an 802.11 Ad Hoc network on the Linux-based SBCs, providing flexibility to adjust device drivers

[4]https://youtu.be/MTwWnZG8wUY?si=MF3wNg9aAs3Mc5mi



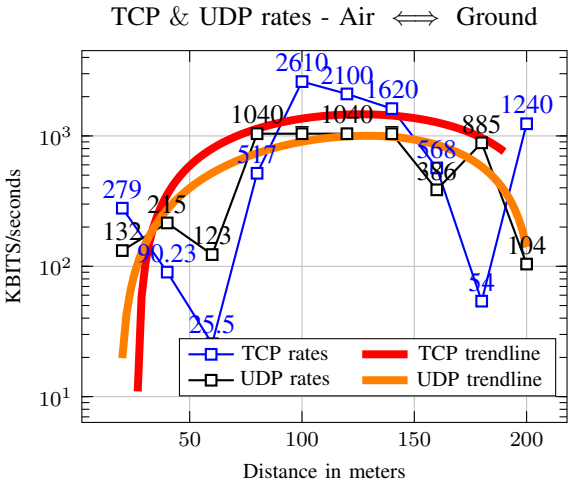
(a) TCP transfer meanwhile getting distance

hfill

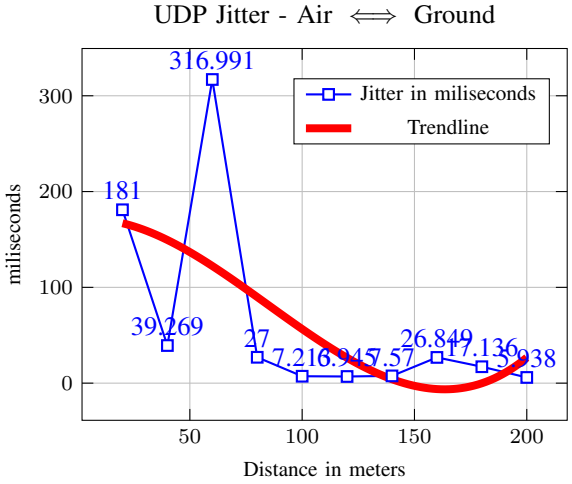


(b) UDP meanwhile getting distance

Fig. 3: Networking evaluation with two UAVs getting far away from each other.



(a) TCP transfer meanwhile getting distance



(b) UDP meanwhile getting distance

Fig. 4: Networking evaluation with one UAVs getting far away from a ground node.

as needed. This setup enabled an IP-based network. The experiment was designed to evaluate data traffic between two drones at increasing distances. One drone was stationary at a height of 10 meters, while the other moved away in 20-meter increments, reaching a maximum distance of 200 meters. At each stop, the moving drone conducted tests using the default parameters of the iPerf3 tool[5] with standard configuration. Figure 3 presents the results.

The primary focus of the observation was on transfer rates using TCP. As expected, transfer rates declined as the distance between the drones increased. While this may not pose an issue for applications that do not require large data transfers, such as images or audio, it could be problematic for applications requiring higher link reliability. For such cases, using UDP with flow control can offer greater resilience as packet loss increases. By limiting the UDP transmission rate to 1 Mbit/s, a stable link was maintained up to a distance of 200 meters.

[5]https://iperf.fr/

B. 802.11 Ad Hoc - Ground-Air

In this section, the tests from the previous experiment involved a fixed node positioned 2 meters above the ground, with a drone flying at an altitude of 20 meters. The drone moved away in increments of 20 meters, conducting tests at each interval. The results are presented in Figure 4.

The experiment first analyzes the TCP and UDP transmission rates, shown in subfigure (a). Unlike the previous

experiment, where both protocols exhibited distinctly asymptotic behaviors, this experiment shows that they demonstrate strikingly similar patterns.

In subfigure (b), the UDP jitter also follows a comparable asymptotic trend. In contrast, subfigure (c) reveals that UDP packet loss is initially high when the nodes are close but rapidly decreases as the distance increases, continuing to fall until communication is no longer possible.

C. Findings

The experiments conducted clearly demonstrated that, as expected, Air-to-Air communication outperforms Air-to-Ground communication. However, it is not as straightforward in design. A key observation is that the range achieved in both scenarios was approximately 200 meters. It is important to highlight that in Air-to-Ground communication, a reasonably direct line of sight between the drone and the ground station was maintained.

Examining Figure 3(a), it is evident that TCP communication achieves a higher transfer rate compared to UDP rates, which might appear counterintuitive. Nevertheless, UDP communication exhibited greater stability.

Analyzing Figure 4(a), TCP communication achieves higher transfer rates than UDP only in certain cases, while the stability of both protocols appears to be relatively similar.

Looking at Figure 3(b) and 4(b), one can observe nearly identical asymptotic behaviors for UDP jitter. However, the jitter results are up to five times worse in Air-to-Ground communication compared to Air-to-Air communication.

The local maximum transfer rate values differ by only 25 meters when comparing the Air-to-Air and Air-to-Ground scenarios. For UDP communication, particular caution is advised, as the results vary significantly in magnitude between the two configurations.

Thus, the region between 100 and 150 meters emerges as the optimal zone for multipurpose communication with this UAV setup, which is considered among the simplest, most accessible, and widely used configurations in academic research.

VI. Conclusions

The GrADyS Framework v2 was further developed by integrating simulations and field experimentation in FANETs and ground devices, bridging the practical and theoretical gaps. We found important trade-offs between range, stability, and throughput in tests with 802.11 ad hoc networks, demonstrating reliable performance up to 200 m. The importance of demonstrating models in realistic environments was clearly established, as models often need calibration. This work combines inexpensive hardware and scientific rigor, improving the reliability, scalability, and reusability of ad hoc flying systems. These developments advance the understanding of UAV-to-UAV and UAV-to-ground communications and more efficient distributed networks and autonomous systems.

This study was financed in part by AFOSR grant FA9550-23-1-0136.

References

[1] M. E. Bruno José Olivieri de Souza, "Ventures, Insights, and Ponderings from Real-World Experiments with FANETs and UAVs," in *29th IEEE Symposium on Computers and Communications (IEEE ISCC 2024)*, 2024, p. to appear.

[2] B. J. O. De Souza, T. Lamenza, M. Paulon, V. B. Rodrigues, V. C. A. Carneiro, and M. Endler, "Collecting Sensor Data from WSNs on the Ground by UAVs: Assessing Mismatches from Real-World Experiments and Their Corresponding Simulations," *Proceedings - IEEE Symposium on Computers and Communications*, vol. 2023-July, pp. 284–290, 2023.

[3] T. Lamenza, J. Kamysek, B. J. O. de Souza, and M. Endler, "Developing Algorithms for the Internet of Flying Things Through Environments With Varying Degrees of Realism," *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pp. 33–40, may 2024. [Online]. Available: https://sol.sbc.org.br/index.php/sbrc_estendido/article/view/29952

[4] T. Lamenza, M. Paulon, B. Perricone, B. Olivieri, and M. Endler, "GrADyS-SIM – A OMNET++/INET simulation framework for Internet of Flying things," in *SBRC 2022 - Brazilian Symposium on Computer Networks and Distributed Systems*, Fortaleza, 2022. [Online]. Available: https://sbrc2022.sbc.org.br/salao-de-ferramentas/

[5] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles," *Springer Proceedings in Advanced Robotics*, vol. 5, pp. 621–635, 2018. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-67361-5_40

[6] S. Baidya, Z. Shaikh, and M. Levorato, "FlynetSim: An open source synchronized UAV network simulator based on ns-3 and ardupilot," *MSWiM 2018 - Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 37–45, oct 2018. [Online]. Available: https://doi.org/10.1145/3242102.3242118

[7] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "ArduSim: Accurate and real-time multicopter simulation," *Simulation Modelling Practice and Theory*, vol. 87, pp. 170–190, sep 2018.

[8] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robotics and Automation Magazine*, vol. 17, no. 3, pp. 56–65, sep 2010.

[9] A. I. Hentati, L. Krichen, M. Fourati, and L. C. Fourati, "Simulation Tools, Environments and Frameworks for UAV Systems Performance Analysis," *2018 14th International Wireless Communications and Mobile Computing Conference, IWCMC 2018*, pp. 1495–1500, aug 2018.

[10] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena," *Mechatronics*, 2014.

[11] N. Sydney, S. Napora, and D. A. Paley, "A multi-vehicle testbed for underwater motion coordination," in *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*, ser. PerMIS '10. New York, NY, USA: ACM, 2010, pp. 107–111. [Online]. Available: http://doi.acm.org/10.1145/2377576.2377597

[12] H. Cabrita and B. Guerreiro, "NOVA.DroneArena: design and control of a low-cost drone testbed," *Proceedings - 2021 International Young Engineers Forum in Electrical and Computer Engineering, YEF-ECE 2021*, pp. 20–25, 2021.

[13] I. Bekmezci, I. Sen, and E. Erkalkan, "Flying ad hoc networks (FANET) test bed implementation," *RAST 2015 - Proceedings of 7th International Conference on Recent Advances in Space Technologies*, pp. 665–668, aug 2015.

[14] B. Perricone, T. Lamenza, M. Paulon, B. J. O. de Souza, and M. Endler, "GrADyS-GS – A ground station for managing field experiments with Autonomous Vehicles and Wireless Sensor Networks," *arXiv*, apr 2022. [Online]. Available: https://arxiv.org/abs/2204.00488v1