Open-Source Quadcopter for Autonomous UAV Networks for Surveillance using Zigbee Radios and Thermal Imaging

Justin Hynes, Alejandro Marcovich, Gustavo Vejarano
Department of Electrical and Computer Engineering
Loyola Marymount University
Los Angeles, USA
jhynes5@lion.lmu.edu, amarcovi@lion.lmu.edu, Gustavo.Vejarano@lmu.edu

MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM) | 979-8-3503-7423-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/MILCOM61039.2024.10773795

Abstract—**This paper details the development of an open-source quadcopter prototype for UAV-to-UAV communication via a Zigbee network. The prototype integrates a thermal imaging system for surveillance and autonomy to fly to destinations determined onboard without ground control. The prototype is assembled on a Parrot Anafi AI quadcopter utilizing a Raspberry Pi Zero 2 W, an XBee transciever, and a TOPDON TC001 thermal camera. UAV operations are executed onboard via services of the Air SDK open-source architecture and implemented in Python and C++. Ground control of the UAV runs on a separate WiFi network, and it overrides the autonomous operation for safety. Flight tests demonstrate successful UAV-to-UAV communication links of up to 113 meters long in an urban environment. Periodic acquisition of thermal images along with corresponding telemetry data (i.e., altitude, geo-location, pitch, yaw, roll) were confirmed. A separate service controls the flight trajectory, which can be determined as a result of onboard image and telemetry data processing in coordination with other equally equipped UAVs.**

Index Terms—**UAV network, thermal imaging, surveillance, autonomy**

I. Introduction

Unmanned aerial vehicles (UAVs) have gained significant traction in both research, commercial, and military realms owing to their flexibility, cost-effectiveness, and diverse application spectrum. A key challenge is the limitation associated with the operational capacity of a single UAV. It may excel in specific and constrained tasks, but its capabilities can be expanded by enabling autonomy and communication with other UAVs.

This paper documents the development and experimental evaluation of an open-source quadcopter for the implementation of autonomous UAV networks within the context of surveillance using thermal imaging. The mission considered in our application scenario is monitoring of small wildfires [1], hence the addition of thermal imaging to our quadcopter.

The problem of UAV-to-UAV commmunications and networks has been surveyed in [2]–[8]. UAV networks reported in these surveys have been implemented and evaluated via simulation in most cases [9]–[15]. Testbeds and prototypes are limited to a few instances to the best of our knowledge [16]–[21], and these do not include autonomy but ground control. This paper bridges the gap with a UAV implementation that adds autonomy.

Our objective is to contribute an open-source quadcopter [22], [23] that includes capabilities to network with other UAVs via UAV-to-UAV direct communication and to determine its flight trajectory onboard without ground control. The quadcopter also collects telemetry data and thermal images. Telemetry data includes altitude, geo-location, and quadcopter's and gimbal's orientations (i.e., pitch, yaw, roll). The quadcopter has onboard processing capability to perform data and image processing to calculate the flight trajectory while communicating with other quadcopters connected to the UAV network to accomplish their mission.

The contributions of this paper are the following:

- An open-source UAV platform for a Flying Ad-Hoc Network (FANET), thermal imaging, and flight autonomy without ground control is made available to the public [22], [23].
- Field tests demonstrate successful FANET connectivity with wireless links of up to 113 meters.
- Field tests demonstrate successful thermal image acquisition along with corresponding telemetry data including UAV's altitude, geo-location, and pitch, roll, yaw of both UAV and gimbal.
- Field tests demonstrate successful execution of waypoints and cardinal direction calculated while flying and onboard to enable full autonomy

The organization of the paper is as follows. The related work is presented in Section II. Section III describes the UAV system architecture. Section IV covers the services that control the UAV operations. Experimental results are presented in Section V. The paper is concluded in Section VI.

II. Related Work

An emerging concept in the domain of UAV communications is FANETs [5]. Several studies, including [16]–[21], report on testbeds and prototypes for FANETs. A two-hop relay network was implemented using Ar.drone 2.0 quadcopters and Raspberry Pis (RPi) in [16]. Though the Ar.drone 2.0 has built-in Wi-Fi, it required an RPi for ad-hoc networking. The RPi

TABLE I: UAV Platforms and Thermal Cameras

Reference	UAV Platform	Thermal Camera
This paper	Parrot Anafi AI	TOPDON TC001
[24]	AggieAir-TIR Custom fixed-wing	Photon 320
[25]	Oktokopter MK-Okto	NEC F30 IS
[26]	DJI Phantom 4 pro	FLIR
[27]	Vulcan hexacopter	FLIR SC305
[28]	DJI M600 Pro hexacopter	FLIR Vue Pro R 640
[29]	ALTUS II fixed-wing	AIRDAS

included two WiFi interfaces. One interface connects to the UAV's WiFi network, and the other interface is configured in ad-hoc mode. UAVs relayed commands from a ground control station (GCS) to extend its reach. The same UAV and RPi configuration was also reported in [17]. The prototype in [18] used an Arduino Wecoms R1 D1 connected via serial communication to a quadcopter, and a LoRa Shield v1.4 transciever connected to the Arduino. The LoRaWAN enables direct UAV-to-UAV communication. The prototype was also evaluated with a two-hop relay network.

Another prototype was reported in [21] that introduced the idea of a leader UAV that maintained communication with the GCS via relay UAVs. As the leader UAV flies further, the GCS commands a relay UAV to take off and serve as relay node in the ad-hoc network. The GCS keeps on adding relay UAVs to the network one by one as the leader travels farther away. The positioning of the relay nodes is determined from the received signal strength indicator to guarantee reliable wireless links. The UAVs were Beopop2 quadcopters with their WiFi module configured in ad-hoc mode. Tests demonstrated operation of the network for up to three hops (i.e., two relay UAVs and the leader UAV).

Another approach for testbeds of UAV networks is the use of motion capture to control UAV operations [19], [20]. Motion-capture cameras track reflective markers placed on UAVs to determine their location and orientation, and a GCS processes these data to command the UAVs' maneuvers. However, flight operations are indoors only due to the motion-capture system. Also, the emphasis in [19], [20] is on centralized flight control from the GCS, not on a FANET.

Other influential contributions to UAV networks address other problems including integration with cellular networks [9], relay chains/trees [10], routing protocols [11], self-recovery after UAV failures [12], flight formation [13], and UAVs serving as base stations in a cellular network [14], [15]. However, experimental results do not include tests on the field but simulations instead.

Thermal imaging and autonomy are also considered in this paper. Current and past uses of thermal cameras on UAVs are implemented on either custom-made platforms [24], [25] or platforms that are not open source [26]–[29]. These platforms do not consider autonomous flights either, but ground control only. Table I lists all UAV platforms and thermal cameras of [24]–[29].

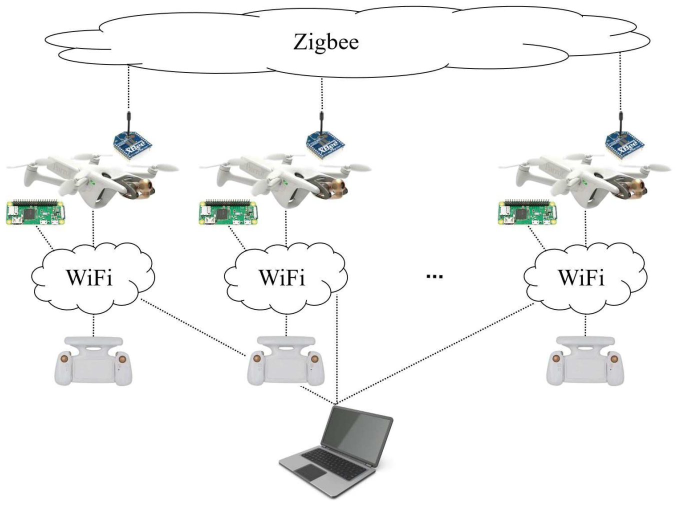


Fig. 1: Network Architecture

III. UAV System Architecture

The network architecture is shown in Fig. 1. Each UAV is a WiFi access point; its controller and RPi connect to this network. The RPi is for acquisition of thermal images. There is also a laptop computer that connects to one of the UAV WiFi networks at a time. It is used for software development, programming of the UAVs and RPis, and for starting/ending the UAV network operations. The controller is for manual command/control of the UAV. The controllers are used only for overriding autonomous operations via manual commands when there are unexpected UAV maneuvers due to bugs in the source code during development. Each UAV has an XBee transceiver to eneble direct UAV-to-UAV connectivity via a Zigbee ad-hoc network.

XBee transcievers are configured using the XCTU software [30] in the *ZigBee Router API* mode within the same *PAN ID* network, enabling them to communicate with each other.

The hardware configuration of a UAV is shown in Fig. 2. It includes a Raspberry Pi Zero 2, a TOPDON TC001 thermal Camera, an XBee transciever, a USB hub, a micro USB to USB-A adapter, a USB-C to USB-A adapter, and a micro USB cable. The USB hub is connected to the drone's USB-C port via the USB-C to USB-A adapter, while the Raspberry Pi's micro USB power port and the XBee module connect to the USB hub. The thermal camera connects to the Raspberry Pi's micro USB data port via the micro USB to USB-A adapter. The UAV's USB C port has a maximum output current of 500 mA, which is not exceeded in our configuration. All components were attached to the UAV using Velcro.

Parrot Air SDK 7.7 [31] is used for software development. The SDK was installed on an Ubuntu OS computer 20.04 LTS. Fig. 3 shows the software architecture of the UAV. The operating system of the UAV is Linux BusyBox, a Linux OS version for embedded systems [32]. The onboard software has three main customizable processes: flight supervisor, guidance, and services. Services are computer programs that are required

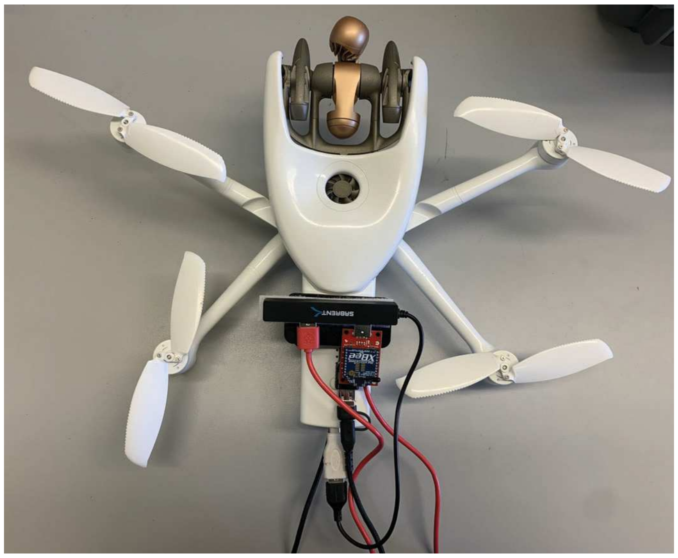


Fig. 2: UAV Prototype

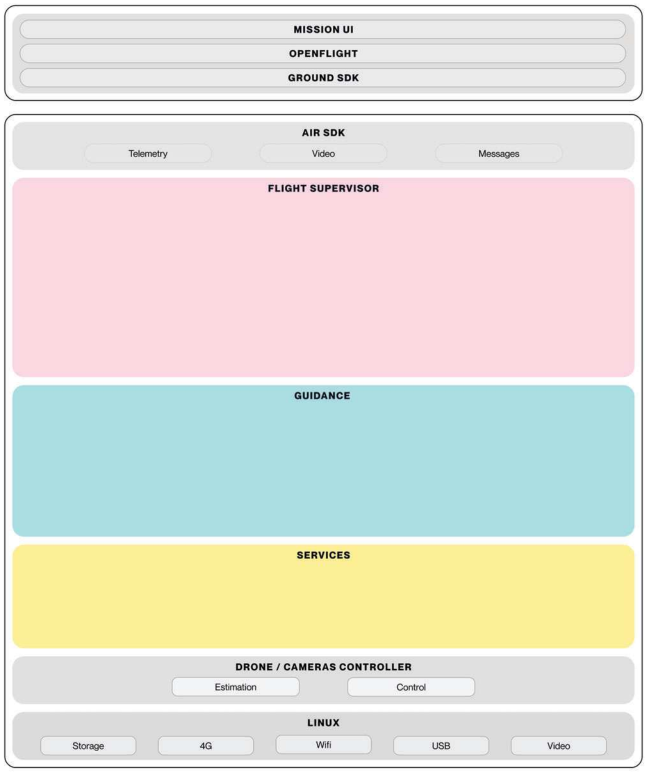


Fig. 3: Software Architecture [31]

for specific guidance. They can also be computer vision algorithms, neural networks, or communication protocols. Services are run on separate Linux processes. Our project includes three custom services only. One service is for acquisition of thermal images and telemetry data. Another service is for generating the next waypoint and orientation that the UAV needs to fly to. The last service is for executing the flight trajectory to reach the new waypoint and orientation.

Detailed information on the installation and operation of this Air SDK mission can be found here [22], [23].

IV. Services

Air SDK allows the development of missions following the architecture in Fig. 3. In our system, mission development is completed in the laptop computer in Fig. 1. The mission needs to be built in this computer and then installed on the UAV via its WiFi network. The UAV restarts after the mission installation, and the mission is activated.

Our goal in this paper is to provide an open-source framework for establishing a FANET via the Zigbee network in Fig. 1, collect telemetry data, capture thermal images, and command the UAV flight trajectory. The trajectory can be calculated onboard after processing of telemetry data and thermal images as well as after communicating with neighbor UAVs. We facilitate this framework with three Air-SDK services that perform the following tasks periodically. All logging of data takes place on a local directory of the UAV.

1) Obtain and log telemetry data including altitude, geo-location, UAV's and gimbal's pitch, yaw, roll
2) Command RPi to capture a thermal image
3) Command RPi to transfer the image over WiFi to the UAV's local directory
4) Transmit telemetry data to neighbor UAVs
5) Determine next waypoint and orientation
6) Go to next waypoint with target orientation

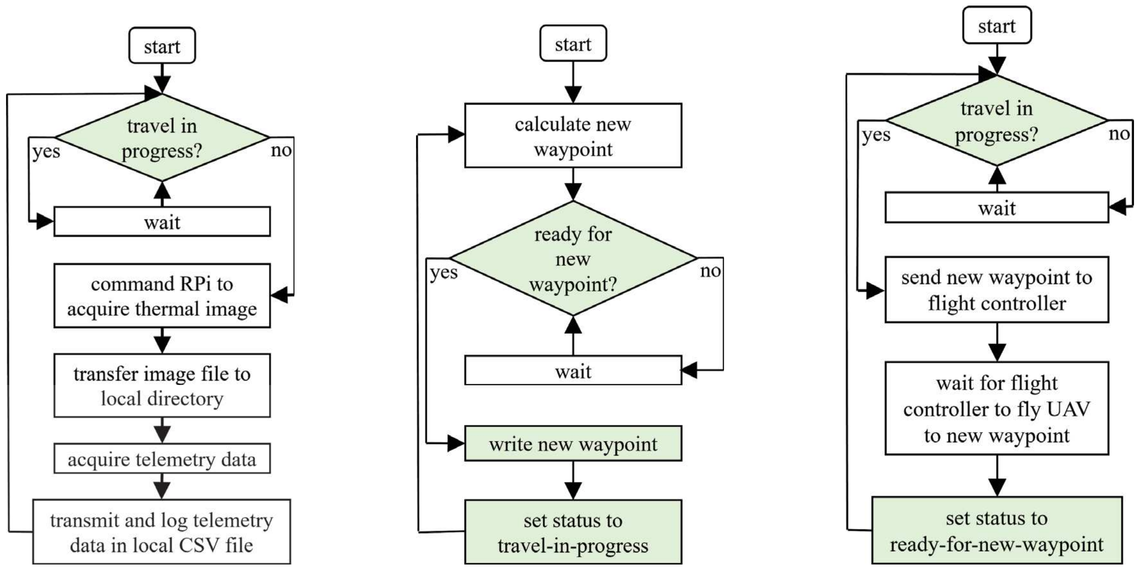
One service takes care of tasks 1 to 4, and it is written in Python. We refer to it as the *telemetry-broadcast* service. Another service takes care of task 5, and it is written in Python too. We refer to it as the *trajectory-determination* service. Lastly, task 6 is executed by another service written in C++. We refer to it as the *go-to* service. Selection of the programming language was determined from Air-SDK template services.

The telemetry-broadcast service is the producer of data, and the trajectory-determination service is the consumer. It can processes thermal images, telemetry data, and data from neighbor UAVs to determine the flight trajectory. Given that the mission of our UAV network (i.e., small wildfire monitoring) is out of scope of this paper, our source code for the trajectory-determination service is only a template that executes a predetermined trajectory, but all data logs and images are available to it in the UAV's local directory for processing.

Services run as separate Linux processes, and our services communicate with each other via temporary files. Air SDK does provide other facilities known as *messages* for this purpose, but we have been unable to configure them properly. Fig. 4 shows the flowcharts of the three services explained next. Blocks highlighted in green indicate access to the temporary files for services to coordinate their actions.

A. Telemetry-Broadcast Service

The service depends on the following Python libraries: paramiko, scp, pyserial, XBee, digi-xbee. Paramiko is a an


start
travel in progress?
yes
no
wait
command RPi to acquire thermal image
transfer image file to local directory
acquire telemetry data
transmit and log telemetry data in local CSV file
start
calculate new waypoint
ready for new waypoint?
yes
no
wait
write new waypoint
set status to travel-in-progress
start
travel in progress?
yes
no
wait
send new waypoint to flight controller
wait for flight controller to fly UAV to new waypoint
set status to ready-for-new-waypoint

(a) Telemetry-Broadcast Service
(b) Trajectory-Determination Service
(c) Go-To Service

Fig. 4: Flowcharts of services

implementation of the Secure Shell 2.0 (SSHv2) protocol. SCP uses a paramiko transport to send and receive files via the Secure Copy Protocol (SCP). Paramiko and SCP are needed to command the RPi to capture thermal images and transfer them to the UAV's local directory via the UAV's WiFi network. Pyserial is for accessing the USB serial port that the XBee radio connects to. The XBee and digi-xbee libraries are used for operating the XBee transceiver including transmission to and reception from neighbor UAVs.

The service logs telemetry data in a comma-separated-values file and image files that all reside in the same local directory. The data entries and image files are time-stamped to indicate correspondence with one another. These data are all available to the next service, trajectory-determination service, in the UAV's local directory.

The telemetry-broadcast service is modeled after a sample service from Air SDK, *telemetry_py*. The source code is available and documented in our repository [22].

The RPi includes a Python script that captures one thermal image every time is run. This script is an adaptation from [33]. The service runs the script and transfers the image from the RPi's local directory to the UAV's local directory using SSHv2 and SCP. The script is available in our repository [22].

B. Trajectory-Determination Service

The service uses two temporary files to coordinate the execution of the flight trajectory. The service runs on an infinite loop that brakes to land the UAV after mission completion. For each iteration, a new waypoint is determined first. The waypoint value can be determined after processing local telemetry data and thermal image and after communicating with neighbor UAVs. The exact calcuation of the new waypoint depends on the application scenario (e.g., wildfire monitoring, wildlife/people surveillance, etc.). Then, the status of the UAV is checked. The status is binary and indicated in one of the temporary files. It can be *ready-for-new-waypoint* or *travel-in-progress*. This file is accessed by both trajectory-determination and go-to services. The trajectory-determination service waits for the status to become *ready-for-new-waypoint* to send the new waypoint to the go-to service. After sending it, the service updates the status to *travel-in-progress*. A waypoint consists of three distances and UAV's orientation with respect to the current location and orientation. There are three distances for x, y, z in meters and a heading value in radians for the change in yaw (i.e., cardinal orientation). The new waypoint is sent to the go-to service via the second temporary file.

C. Go-To Service

The service checks for the status file to become *travel-in-progress*. If it does, it sends a *move_by* command to the flight supervisor (Fig. 3) with the new waypoint value. If it does not, it sends a *move_by* command to the flight supervisor with a null waypoint to stay hovering where it is. The service receives an event from the flight supervisor when the *move_by* command has been executed. This is indication that the UAV reached the waypoint value. The service updates the value of the status file to *ready-for-new-waypoint* when the event is received. The go-to service is modeled after a sample service from Air SDK, *move_along*. The source code of both trajectory-determination and go-to services is available and documented in our repository [23].

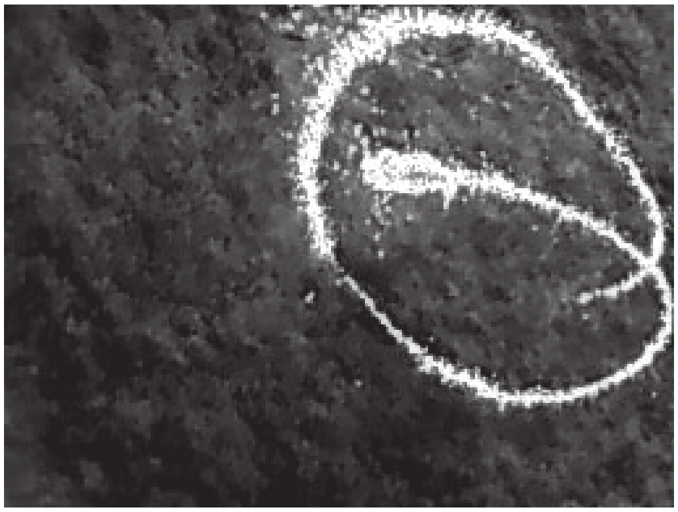


Fig. 5: Sample Image from Thermal Camera

V. Experimental Results

There were three tests of our prototype to verify successful operation of our services including (1) flight-trajectory determination and execution, (2) aquisition of thermal images, and (3) UAV network.

In the first test, the UAV flew indoors while executing a flight trajectory that included rotations along the Z axis only (i.e., yaw). A video of this test is publicly available [34]. The video shows the UAV taking off and executing the flight trajectory determined onbard by the trajectory-determination service. Although not shown in the video, the UAV successfully logged telemetry data and thermal images in its local directory. It also transmitted the telemetry data periodically.

In the second test, hot water was poured over grass, and the thermal camera captured the heat as shown in Fig. 5.

The third and final test assessed connectivity between two XBee transceivers as the distance between them increased and under two different scenarios, open field vs. field surrounded by buildings. There were three trials conducted with trials 1 and 2 being in the open field and trial 3 being surrounded by buildings. The starting distance between the two XBee transceivers was 91.4 meters. The received packet trace was recorded for 5 minutes to calculate the average packet delivery rate (PDR). The distance was then increased by 3 meters, and the PDR was calculated again. The source code for this test is available in our repository [35].

Trial 1 was conducted at the campus of Loyola Marymount University (LMU) at a location named *Sunken Garden*. This site was ideal due to its lack of nearby buildings, low pedestrian traffic, and large open fields. Trial 2 was also conducted at Sunken Garden, but this time, an on-campus event resulted in fences blocking half of the area, limiting the measurement distance to 15.2 meters in addition to the initial 91.4 meters. Trial 3 took place in front of Pereira Hall, another location on the LMU campus. This site is characterized by the presence of nearby buildings, high foot traffic, and vehicular movement.

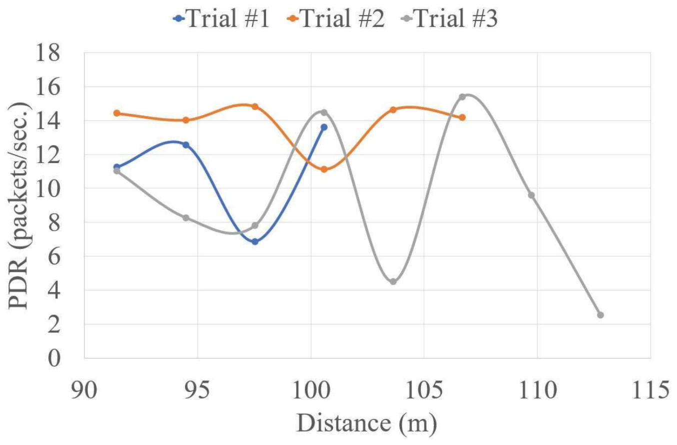


Fig. 6: Packet Delivery Rate

Trial 1 in Fig. 6 shows higher and more consistent PDR as the distance increases. Trials 2 and 3 show higher variability of the PDR with distance and, for Trial 3, a rapid decreases after 113 meters. These results are expected from the type of environments being tested. The oscillatory variation of the PDR with distance is attributed to multi-path interference.

VI. Conclusions and Future Work

This paper explored the integration of thermal imaging, telemetry data acquisition, UAV-to-UAV communication, and autonomous flight operation within a Parrot Anafi AI quadcopter. Thermal imaging was implemented with a TOPDON TC001 thermal camera connected to a Raspberry Pi Zero 2 W. UAV-to-UAV communication was implemented via a Zigbee network of XBee transcievers connected to UAVs. Flight autonomy was implemented via services in the software architecture of the UAV. These services are individual Linux processes that communicate with each other via temporary files. All the development, dependencies, and source code are documented and available to the public in our repositories [22], [23]. The combination of these technologies can help contribute to the critical demand for accessible, efficient, and customizable autonomous UAV networks for surveillance using thermal imaging. The findings highlight the feasibility of employing lightweight, cost-effective components in complex applications, paving the way for broader adoption and further innovation. The successful validation of this system through testing and practical deployment underscores its potential for remote sensing and monitoring.

Future work includes extended testing of the UAV network including multiple hops and development of our mission on monitoring small wildfires [1]. The mission involves thermal-image registration from adjacent UAVs and autonomous determination of flight trajectories using image processing and coordination among UAVs.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 2301707.

References

[1] G. Vejarano. Eri: Fault-tolerant monitoring of moving clusters of targets using collaborative unmanned aerial vehicles. [Online]. Available: https://www.nsf.gov/awardsearch/showAward?AWD_ID=2301707

[2] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1123–1152, 2016.

[3] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2624–2661, 2016.

[4] X. Cao, P. Yang, M. Alzenad, X. Xi, D. Wu, and H. Yanikomeroglu, "Airborne communication networks: A survey," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1907–1926, 2018.

[5] M. A. Khan, A. Safi, I. M. Qureshi, and I. U. Khan, "Flying ad-hoc networks (fanets): A review of communication architectures, and routing protocols," in *2017 First International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT)*, 2017, pp. 1–9.

[6] S. Al-Emadi and A. Al-Mohannadi, "Towards enhancement of network communication architectures and routing protocols for fanets: A survey," in *2020 3rd International Conference on Advanced Communication Technologies and Networking (CommNet)*, 2020, pp. 1–10.

[7] A. Sharma, P. Vanjani, N. Paliwal, C. M. Basnayaka, D. N. K. Jayakody, H.-C. Wang, and P. Muthuchidambaranathan, "Communication and networking technologies for uavs: A survey," *Journal of Network and Computer Applications*, vol. 168, p. 102739, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520302137

[8] A. I. Hentati and L. C. Fourati, "Comprehensive survey of uavs communication networks," *Computer Standards & Interfaces*, vol. 72, p. 103451, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548919303411

[9] M. M. Azari, G. Geraci, A. Garcia-Rodriguez, and S. Pollin, "Uav-to-uav communications in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 9, pp. 6130–6144, 2020.

[10] P.-M. Olsson, J. Kvarnström, P. Doherty, O. Burdakov, and K. Holmberg, "Generating uav communication networks for monitoring and surveillance," in *2010 11th International Conference on Control Automation Robotics & Vision*, 2010, pp. 1070–1077.

[11] X. Tan, Z. Zuo, S. Su, X. Guo, X. Sun, and D. Jiang, "Performance analysis of routing protocols for uav communication networks," *IEEE Access*, vol. 8, pp. 92 212–92 224, 2020.

[12] G.-H. Kim, I. Mahmud, and Y.-Z. Cho, "Self-recovery scheme using neighbor information for multi-drone ad hoc networks," in *2017 23rd Asia-Pacific Conference on Communications (APCC)*, 2017, pp. 1–5.

[13] J. Xiangyu, W. Sentang, L. Xiang, D. Yang, and T. Jiqiang, "Research and design on physical multi-uav system for verification of autonomous formation and cooperative guidance," in *2010 International Conference on Electrical and Control Engineering*, 2010, pp. 1570–1576.

[14] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-uav enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[15] Z. Wang, L. Duan, and R. Zhang, "Adaptive deployment for uav-aided communication networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4531–4543, 2019.

[16] G.-H. Kim, J.-C. Nam, I. Mahmud, and Y.-Z. Cho, "Multi-drone control and network self-recovery for flying ad hoc networks," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2016, pp. 148–150.

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

[18] Q. Zirak, D. Shashev, and S. Shidlovskiy, "Swarm of drones using lora flying ad-hoc network," in *2021 International Conference on Information Technology (ICIT)*, 2021, pp. 400–405.

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

[20] J. P. HOW, B. BEHIHKE, A. FRANK, D. DALE, and J. VIAN, "Real-time indoor autonomous vehicle test environment," *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, 2008.

[21] O. Shrit, S. Martin, K. Alagha, and G. Pujolle, "A new approach to realize drone swarm using ad-hoc network," in *2017 16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2017, pp. 1–5.

[22] J. Hynes and G. Vejarano. Anafi ai service for zigbee radio, telemetry, and thermal imaging. [Online]. Available: https://github.com/Intemnets-Lab/uav_network/tree/main/missions/telemetry_thermal_versions

[23] A. Marcovich and G. Vejarano. Anafi ai service for setting uav's flight trajectory. [Online]. Available: https://github.com/Intemnets-Lab/uav_network/tree/main/missions/moveby

[24] H. Sheng, H. Chao, C. Coopmans, J. Han, M. McKee, and Y. Chen, "Low-cost uav-based thermal infrared remote sensing: Platform, calibration and applications," in *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, 2010, pp. 38–43.

[25] J. Bendig, A. Bolten, and G. Bareth, "Introducing a low-cost mini-uav for thermal- and multispectral-imaging," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XXXIX-B1, pp. 345–349, 2012. [Online]. Available: https://isprs-archives.copernicus.org/articles/XXXIX-B1/345/2012/

[26] G. Messina and G. Modica, "Applications of uav thermal imagery in precision agriculture: State of the art and future research outlook," *Remote Sensing*, vol. 12, no. 9, 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/9/1491

[27] W. H. Maes, A. R. Huete, and K. Steppe, "Optimizing the processing of uav-based thermal imagery," *Remote Sensing*, vol. 9, no. 5, 2017. [Online]. Available: https://www.mdpi.com/2072-4292/9/5/476

[28] V. Sagan, M. Maimaitijiang, P. Sidike, K. Eblimit, K. T. Peterson, S. Hartling, F. Esposito, K. Khanal, M. Newcomb, D. Pauli, R. Ward, F. Fritschi, N. Shakoor, and T. Mockler, "Uav-based high resolution thermal imaging for vegetation monitoring, and plant phenotyping using ici 8640 p, flir vue pro r 640, and thermomap cameras," *Remote Sensing*, vol. 11, no. 3, 2019. [Online]. Available: https://www.mdpi.com/2072-4292/11/3/330

[29] V. G. Ambrosia, S. S. Wegener, D. V. Sullivan, S. W. Buechel, S. E. Dunagan, J. A. Brass, J. Stoneburner, and S. M. Schoenung, "Demonstrating uav-acquired real-time thermal data over fires," *Photogrammetric Engineering & Remote Sensing*, vol. 69, no. 4, pp. 391–402, 2003.

[30] Digi. Next generation configuration platform for xbee/rf solutions. [Online]. Available: https://www.digi.com/products/embedded-systems/digi-xbee/digi-xbee-tools/xctu

[31] Parrot. Air sdk documentation. [Online]. Available: https://developer.parrot.com/docs/airsdk/index.html

[32] E. Andersen and D. Vlasenko. Busybox: The swiss army knife of embedded linux. [Online]. Available: https://busybox.net/about.html/

[33] L. Wright. (2023) Pythermalcamera. [Online]. Available: https://github.com/leswright1977/PyThermalCamera

[34] J. Hynes. (2024) Parrot anafi ai drone indoor test flight with integrated thermal module mounted. [Online]. Available: https://www.youtube.com/watch?v=eZZYy90ekqE

[35] ——. (2023) Packet rate evaluation & packet loss measurement. [Online]. Available: https://github.com/Intemnets-Lab/uav_network/tree/main/missions/packetPerformance_assessment