2020 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. September 1-4, 2020

# HorizonBlock: Implementation of an Autonomous Counter-Drone System

N. Souli, R. Makrigiorgis, A. Anastasiou, A. Zacharia
P. Petrides, A. Lazanas, P. Valianti, P. Kolios, and G. Ellinas

***Abstract*— Unmanned Aircraft Systems (UASs) are technologically advancing at such a rapid pace that domain experts are now highly concerned of the potential misuse of the technology that can be used for unlawful actions with detrimental effects. The most effective measure to counteract the operation of rogue drones are electronic anti-drone systems that in one way or another intercept the normal operation of a rogue agent. In this work we develop an intelligent pursuer drone that implements novel lightweight functions to meet all necessary interception steps (i.e., detection, tracking and interception) in addition to self-localizing using signals of opportunity in order to maintain perception when performing wireless jamming against a rogue drone.**

***Index Terms*— UAS, Counter-Drone, Machine Learning, Computer Vision, GPS jamming**

## I. Introduction

Unmanned Aircraft Systems (UASs) have attracted enormous interest from both the scientific and industrial community due to their potential transformative effect for a great number of application scenarios. Based on the latest figures on UAS-related technology, consumer-drone demand will increase sharply over the next few years with the global market for drone technology reaching 43.1B$ by 2024 [1].

However, the increasing improvement of UAS capabilities with higher levels of intelligence and autonomous features can also potentially introduce new threats to public spaces and critical infrastructures. The anonymous and uncontrolled purchase and use of drone platforms has led to the need for technology-specific security systems to counter potentially malicious actions.

Importantly, there are no sufficient solutions to date to effectively detect, track, and intercept rogue drones in a safe manner. The research community has concentrated on innovative detection techniques, such as RF signal sniffing, sensors, and computer vision [2]. In addition, interception techniques include net-casting, RF denial systems, and high-power lasers [3], [4]. Nevertheless, as mentioned in [5], substantial further work is required for effective UAS interception solutions.

Safety for critical infrastructure systems is crucial as indicated in the recent drone incidents at London's Gatwick and Heathrow airports, where drones flew over restricted airspace, presumably targeting the interruption of air traffic. In those incidents, British police successfully countered the drones, by using wireless jamming to block the UASs' remote control functionality. As reported in [6], jamming is the most effective interception method. However, the use of wireless jamming over extended ranges may severely affect normal operation of systems that depend on such signals as recently highlighted by the Federal Aviation Administration (FAA).

In accordance, this work proposes a fully autonomous aerial counter-drone system which is used to detect, track, and jam a rogue drone. Upon alerted of the target, the counter-drone system takes-off with the aim of detecting and tracking the rogue drone before intercepting its operation via wireless jamming. Importantly, and to maintain navigation in space, the pursuer drone self-localizes using a novel technique based on signals of opportunity as it will be discussed in Section V. Figure 1 depicts the high-level process of our proposed system to aid understanding.

The rest of this work is structured as follows. Related work is included in Section II and details of the four system components that comprise the proposed *HorizonBlock* counter-drone system are included in Section III, Section IV, and Section V, respectively. Section VI elaborates on implementation and integration aspects of the proposed system and details prototype tests. Concluding remarks regarding this work are presented in Section VII.

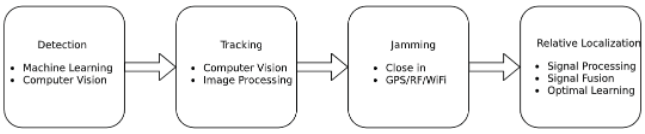



Fig. 1: High level view of the proposed HorizonBlock counter-drone system.

## II. Related Work

Industry leaders in the safety and security sector and organizations that specialize in military and law enforcement technologies, have developed several counter UAS (CUAS) solutions that are capable of detecting, localizing, and neutralizing malicious UAVs. The CUAS solutions currently available in the market can be divided into three main categories as summarized below:

- Stationary CUAS systems: Stationary solutions have increased range of surveillance due to perpetual access to power resources. These solutions have relatively high cost and require power amplifiers that extent the

Nicolas Souli, Rafael Makrigiorgis, Andreas Anastasiou, Angelos Zacharia, Petros Petrides, Alexandros Lazanas, Panayiota Valianti, Panayiotis Kolios, and Georgios Ellinas are with the Department of Electrical and Computer Engineering and the KIOS Research and Innovation Center of Excellence, University of Cyprus, {nsouli02, rmakri01, aanast01, azaxar01, ppetri02, alazan01, valianti.panayiota, pkolios, gellinas}@ucy.ac.cy

neutralization range. This could potentially result in interference with legitimate telecommunication systems as well as pose a risk to public health.

- Portable CUAS systems: Portable solutions have the advantage of mobility. However, due to their small size and simplified architecture, they are not able to autonomously detect and localize UAS; rather, the user has to manually aim towards the malicious UAS. Additionally, the neutralization capabilities of such equipment have very limited range, usually several hundreds of meters.
- Aerial CUAS systems: Aerial solutions are installed on UASs with the ability to patrol a specific area and detect and track any malicious UAS in the vicinity before performing neutralization. This kind of solution requires more complex and sophisticated algorithms. Nevertheless, this solution can result in minimal wireless interference due to the close proximity between the pursuer and rogue drones. Furthermore, the range of such a solution closely relates to the flight characteristics of the pursuer drone and particularly the level of flying autonomy.

Counter UAS technologies have already attracted ample research interest. The various existing works focus primarily on detecting the presence of a rogue drone and developing techniques to take control of it or intercept its operation. Specifically, in [7] an anti-UAS system has been developed that combines multiple passive surveillance technologies to perform detection, localization, and radio frequency jamming for neutralization. A relatively recent work detailed in [9], describes an aerial CUAS for catching a rogue UAV using a net carried and launched by the pursuer UAV. However, this approach assumes that the target has already been detected accurately in terms of position and velocity. In [7] and [8] an RF signal jamming technique has been implemented that interferes with the communication link between the UAV and the ground station resulting in loss of the control link. The approach in [8] uses a radar to detect the target, while techniques developed in [7] use audio signals, computer vision, and RF analysis for the detection of the target.

Other research has focused on restricting the access of a malicious UAS in a specific area by taking advantage of the UAS's subsystems, also known as a man-in-the-middle attack. More specifically, the authors in [10] exploit WiFi vulnerabilities of the UAS communication link to launch network-based attacks. Moreover, [11] describes an attack on the UAS's dynamic state estimation by exploiting the vulnerabilities of common state estimation algorithms to misguide its navigation systems and therefore prevent it from flying inside a restricted area. Another relevant work, presented in [12], considers a swarm of patrolling UAVs positioned in a formation around the malicious drone in such a way to enable the collision avoidance system of the malicious drone and therefore limit its movement. Additionally, the swarm moves accordingly to escort the intruder outside of the restricted area.

Contrary to the aforementioned studies, the proposed system presents an innovative, autonomous, and low-cost approach to counteract the mission of rogue drones in a safe and accurate manner.

## III. Detection and Tracking

Since commercial off-the-shelf UASs have flight times of not more than 30 minutes, scanning an area trying to detect possible threats is not very effective. Hereafter, we assume that a ground system is in place to alert of any intrusion and trigger the pursuer drone activation. Thereafter, the pursuer drone takes on the task of detecting and tracking the target, intercepting it mid-air by closing up and initiating jamming, while ensuring that it maintains situational awareness and perception of the environment using signals of opportunity as it will be discussed in the sequel.

### A. Detection

Drone detection can be achieved using computer vision techniques and/or convolutional neural networks (CNN). In this work, a CNN is utilized in order to precisely detect the target. More specifically, Darknet V3 [13] is used, as Darknet's YOLO network is quite popular for its detection performance, especially when it comes to the smaller version of it, tiny YOLO V3. This one is mostly preferred and well-suited for real-time detection of drones that can travel with a top speed of 70km/h. The first step towards detection is the creation of a dataset and labeling images of drones. Currently, a dataset of approximately 700 drone images is utilized, taken at an approximate distance of up to 30m at different angles (side view, top/bottom view, etc.). Figure 2 illustrates an example of the detection of a drone in an environment which was not provided during training.

Detection was evaluated on a laptop using Intel's i7-7700HQ processor and an Nvidia Geforce GTX 1060 graphics card achieving approximately $15-20$ frames per second (FPS) on a 720p video stream from the UAS. Further, an NVIDIA Jetson Nano embedded on the drone was tested, detecting up to 15 FPS, without skipping any frames, using the latest CUDA-compatible [14] OpenCV's [15] library build.

Fig. 2: Detection of a DJI M210 drone.

### B. Tracking

After detecting the rogue drone, the counter-drone system's tracking algorithm is engaged. In order to achieve fast tracking, the detection's bounding box is utilized. Upon achieving a detection bounding box, a normalization of the

coordinates is performed using the video stream's size ($x$-width, $y$-height). Thereafter, a position correction takes place in order for the pursuer to move in closer to the rogue drone. Initially, the area of the detection box is compared in successive frames and if the area is larger than $10\%$ of the image, it is considered that the target is in very close proximity, forcing the pursuer drone to fall back. On the other hand, if the detection box is smaller than $6\%$ of the image, then the target is considered to be at a distance, thus the pursuer moves closer to the target. Hence, the effective jamming range is set so that the detected bounding box is within $7-10\%$ of the captured image. Clearly, depending on the camera's field of view and the jamming intensity, these parameters can be adjusted accordingly.

With respect to the flight control steps, signals for the yaw and throttle of the pursuer flight controller are provided based on the detected box position and the current pursuer's posture. The thresholds for the yaw and throttle movement signals are set to $0.4-0.6$ (after normalization) of the width and height of the image. Algorithm 1 below provides a high-level description of the steps followed in order to track the rogue drone. Clearly the exact motion turning rates depend on the pursuer drone flight characteristics.

**Algorithm 1** Tracking the rogue drone.

**Input** Camera's video frame

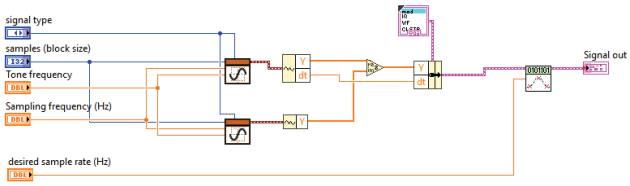
Detect drone in the frame
**if** drone detected **then**
    Normalize box x, y center, based on image size
    **procedure** CHECK BOUNDING BOX POSITION
        **if** box $>$ area threshold max **then**
            signal 'move back' //collision avoidance
        **else if** box $<$ area threshold min **then**
            signal 'move towards'
        **else**signal 'stop moving' //pursuer stopped
        **if** box.x $<$ X threshold min **then**
            signal 'turn anti-clockwise'
        **else if** box.x $>$ X threshold max **then**
            signal 'turn clockwise'
        **else**signal 'stop turning'
        **if** box.y $<$ Y threshold min **then**
            signal 'rise up'
        **else if** box.y $>$ Y threshold max **then**
            signal 'move down'
        **else**stop rising
    Execute turn, rise, move
**else**
    Return to home base

## IV. JAMMING

Jammers emit interference signals at specific wireless bands in an effort to interfere with the communication link between the transmitter and the receiver (e.g. the satellite and GNSS receiver on the drone). Furthermore, these interference techniques typically intend to transmit continuously overpowered radio signals on the GNSS bands forcing the receiver to provide erroneous position, velocity, and time (PVT) information during the navigation process. Currently, this is the most effective wireless interception technique.

In this work, a jamming system is implemented utilizing an SDR (Software Define Radio) and LabVIEW software as shown in Fig. 3. SDR systems are well fitted for this task, since they tend to be robust, reliable, and can transmit various different types of signals. In our case, the LabVIEW software is used to generate the interference baseband signals, and a USRP B200 SDR radio card, paired with an omnidirectional antenna, is transmitting these signals to disrupt the GNSS receiver.


(a) USRP b200




(b) LabVIEW block diagram

Fig. 3: Hardware and software implementation of the jamming system.

To validate the performance of the jamming module, various interference signals were generated and emitted by the USRP B200 in an effort to jam the GPS receiver of a rogue drone. The most common technique is to transmit a continuous wave using amplitude modulated pulses. The various signals that the jammer can generate are described as single-tone, multi-tone (multiple signal waveforms with different frequency components), chirp signals with instantaneous frequency changing over time, and a sinusoidal pulse train. The interference signals are represented in the time and frequency domain as shown in Figs. 4 and 5, respectively.

Using the LabVIEW software tool and the SDR hardware, the continuous wave signals are up-converted to RF GPS frequency bands, specifically in L1 at $1.57542$GHz and transmitted via the antenna.

## V. RELATIVE LOCALIZATION

The proposed counter-drone system can detect, track, and jam a rogue drone. However, the use of a jamming component to neutralize the target can disrupt the GNSS

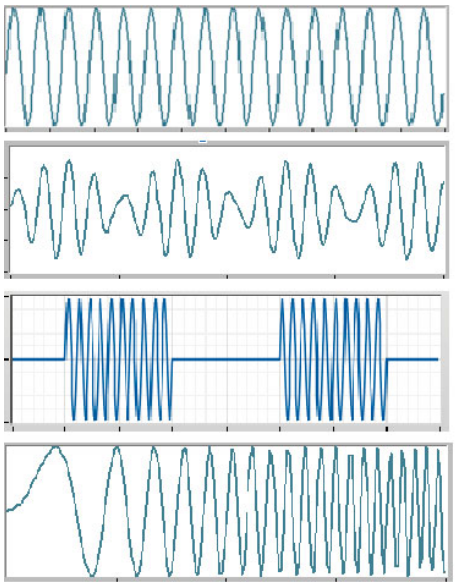


Fig. 4: Time-domain interference signals.

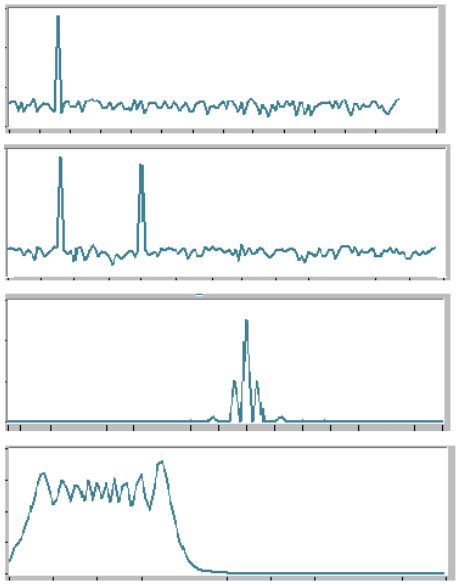


Fig. 5: Frequency-domain interference signals.

signals of the pursuer drone as well. As a mitigation measure, the proposed system incorporates a relative positioning component utilizing signals of opportunity (SOP) in order to self-localize and safely return to an initial position when GNSS signals become unavailable.

### A. Signals Of Opportunity

Clearly, GNSS signals quickly become unavailable in the presence of interference (including jamming or spoofing) [18]–[20] and in deep urban canyons [21]–[23]. In turn, navigation systems utilize an inertial navigation system (INS) [24] and light and range sensors [25] to determine their location in the absence of GNSS signals. The shortfall of these solutions, including data degradation, loss of signal due to multipath and antenna obstruction, has led to the exploration of alternative approaches for localization [21], [26].

Lately, the utilization of SOPs has been introduced as a viable alternative for navigation, when GNSS signals become unstable/unavailable [16], [23], [27]. SOPs (e.g., AM/FM radio, cellular, TV signals, etc.) are fit for navigation purposes because these signals are readily accessible [24], [28].

### B. Relative Positioning Component Framework

This section examines how relative localization can be achieved using signals that are already available in the environment, and derives an online procedure for the exploitation of these signals for positioning in the current GNSS-denied environment. The relative positioning component (RPS) considers the signal characteristics over a large spectrum of frequency bands and derives a tracking algorithm to accurately estimate the pursuer drone's trajectory in space and time using an arbitrary set of unknown reference positions.

Complementary to prior investigations, the RPS achieves relative localization without taking into consideration the location of the transmitters. In addition, it does not employ any GNSS signal information for the relative trajectory extraction and is using only the SOPs measurements. Furthermore, an extended Kalman filter (EKF) approach and an optimal learning (OL) methodology are utilized to improve the performance of the location estimate in an online fashion.

**Algorithm 2** Relative positioning algorithm.

**Input** RSS dataset at each route point

1: Frequency spectrum at the current position of the moving vehicle scanned and relative transmitters set
2: **procedure** RELATIVE POSITION CALCULATION
3: Frequency spectrum analyzed and mean RSS $\theta_k^l$ computed.
4: $\theta_k^l$ utilized to calculate the relative distances using the path-loss model from the transmitters set
5: Multilatteration applied to estimated locations $\widehat{\mathbf{x}^*}^{N-n}$
6: EKF on $\widehat{\mathbf{x}^*}^{N-n}$
7: When the number of samples $N-n$ reach a threshold value OL procedure starts
8: Knowledge gradient (KG) policy applied
9: Best decisions of the frequency feature set $\mathbf{F_k}^{N-n}$ extracted applied on relative position calculation procedure
10: Relative position $x_{k_{ONL}}^{N-n}$ estimated
11: The RPS-OL procedure ends when $n = N$

The proposed RPS algorithm employed by the pursuer drone is shown in Alg. 2. The first step of the algorithm considers the collection of received signal strength data from a particular frequency spectrum. The sweep of the spectrum can be achieved using a software defined radio onboard the pursuer drone. The spectrum sweep $X$ collected at an arbitrary instance $k$ is then aggregated into $l$ blocks and the mean $\theta_k^l$ RSS value of each block $l$ is computed. Using $\theta_k^l$ and a path-loss model to accurately model the signal

propagation in the given environment, then the estimated distances from $l$ reference locations can be derived. In turn, the position of the drone can be extracted using known multilateration techniques as in [30] and [31]. Extended Kalman filtering is then applied to further improve on the location accuracy. To implement this, a measurement and a motion model of the pursuer drone are utilized as follows:

$$x_{k+1} = x_k + T_s \begin{bmatrix} \cos\phi_k & 0 \\ \sin\phi_k & 0 \end{bmatrix} \left( \begin{bmatrix} u_{x_k} \\ u_{y_k} \end{bmatrix} + w_k \right) \quad (1)$$

$$d_{k+1} = \sqrt{(xl - x)^2 + (yl - y)^2} + n_k \quad (2)$$

where $x_k = [x \quad y]^T$ denotes the current position in 2D and $x_{k+1} = f(x_k, u_k)$ denotes the next estimated position calculated using the motion model with:

$$\mathbf{F} = \frac{\partial f}{\partial x_{k+1}} \mid_{x_k,\, u_k} \quad (3)$$

and the measurement model $d_{k+1} = h(x_k, n_k)$ with:

$$\mathbf{H} = \frac{\partial h}{\partial x_{k+1}} \mid_{x_k} \quad (4)$$

where $u_k$ denotes the linear velocity readings used as an input to the motion model, while $\phi_k$ is the heading angle of the moving vehicle. $w_k$ represents the process noise with zero mean normal distribution and covariance $\mathbf{Q} = (0.1)\mathbf{I}_{2x2}$. The measurement model relates to the range between the route points with $xl$ representing the position estimates extracted using the SOPs and $n_k$ the zero mean measurement noise with constant covariance $\mathbf{R} = 0.01$. In the prediction step, the velocity, heading angle readings, and the motion model are utilized to produce a state at a given timestep $k$. Using the measurement model and importing the range measurements $z_{k+1}$ extracted using the SOP values from the previous steps of RPS, the correction step takes place and a relative trajectory is created. The relative coordinates using the EKF are then calculated as follows:

$$x_{k+1} = x_k + \mathbf{K}(z_{k+1} - d_{k+1}) \quad (5)$$

$$\widehat{X}^* = [\, x^1_{k+1},\, x^2_{k+1},\, ...,\, x^F_{k+1}] \quad (6)$$

In addition, and to achieve online execution, the information-rich frequency bands are selected instead of the complete spectrum by periodically evaluating their strength. Toward this end, a reliable and robust online relative localization using an OL technique as discussed in [32] is employed. A ranking problem regarding useful frequencies is solved by employing a linear regression problem to estimate the $\mu^{\pi}_x$ of ground truth values for the OL technique using $\widehat{X}^*$ at a subset of samples $N - n$. As the exact value of the true mean (true position) remains unknown, the following equations are utilized to estimate a set of alternative values that can characterize the ground truth at samples $N - n$:

$$\mu_i = \theta^T X + \epsilon \quad (7)$$

where $\theta$ represents a vector of weights with random initial values, while $X$ is the full frequency dataset. As the experiment repeats itself, the weights converge and the $\mu_i$ estimates of the ground truth values ($\mu^{\pi}_x$) utilizing a decision policy ($\pi$). Using a recursive algorithm (Bayesian) and assuming to have an $\mu^n(x)$ vector of beliefs with covariances $\Sigma^n(x)$ the means and covariances keep updating to calculate the best choices ${\mathbf{F_k}}^{N-n}$ utilizing a knowledge gradient policy (KG) [33], [34].

$$\theta^n = \theta^{n-1} + \frac{1}{\gamma^n}\Sigma^{n-1}x^n\epsilon^n \quad (8)$$

$$\epsilon^n = \mu^n - \theta^{n-1}x^{n-1} \quad (9)$$

$$\Sigma^n = \Sigma^{n-1} - \frac{1}{\gamma^n}(\Sigma^{n-1}x^n(x^n)^T\Sigma^{n-1}) \quad (10)$$

$$\gamma^n = 1 + (x^n)^T\Sigma^{n-1}x^n \quad (11)$$

$$\mu^{n+1}(x) = \Sigma^{n+1}(\Sigma^n\mu^n + W^{n+1}_x) \quad (12)$$

$$W^{n+1}_x = \mu^n + \epsilon^{n+1} \quad (13)$$

The $X^{KG}$ value represents the measurement decisions extracted utilizing $X$; the index of features that can characterize the true mean with the higher accuracy. As a final step, using $X^{KG}$ the ${\mathbf{F_k}}^{N-n}$ is calculated and applied on RPS. The $x^{N-n}_{k_{ONL}}$ iteratively estimated until $n = N$ leading to an online relative trajectory computation problem as follows:

$$\widehat{X}^*_{ONL} = [\, x_{ONL_k},\, x_{ONL_{k+1}},\, ...,\, x_{ONL_{k+N}}] \quad (14)$$

### C. Experimental results

Several field experiments were conducted utilizing various transmission bands and using the proposed RPS-OL in order to validate our approach. A broadband antenna mounted on a SDR module onboard was used to obtain SOP signals across a large frequency spectrum (via HackRF-one software-defined radio). Figure 6 illustrates the hardware and software set-up of the system. The tracking performance of RPS-OL is assessed by calculating the relative trajectory of the moving medium and comparing it with the true target position (GPS trajectory - decimal degrees converted into meters). The deviations between the real and the relative trajectory are illustrated in Fig. 7 and statistical analysis utilizing the distance difference (in meters) starting from an initial point is shown in Fig. 8. It is evident that RPS-OL achieves accurate localization that can be adopted in the absence of GNSS signals (including the situation where there is jamming interference).

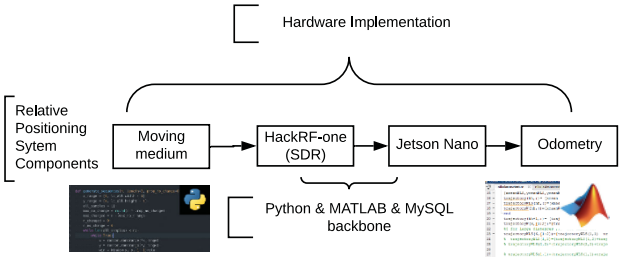



Fig. 6: Block diagram of RPS-OL software and hardware implementation.

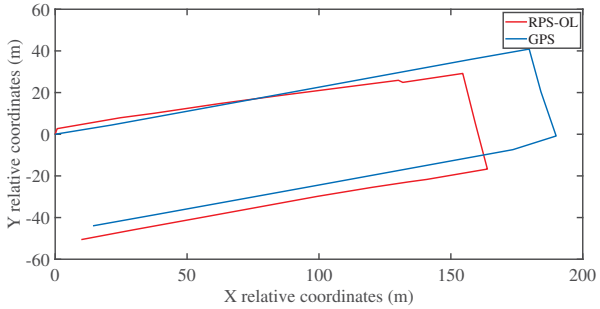


Fig. 7: Relative trajectory compared to GPS trajectory utilizing RPS-OL.

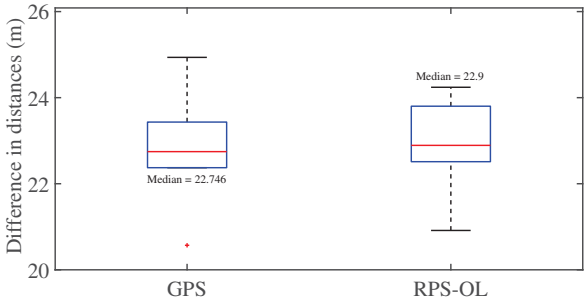


Fig. 8: Statistical analysis of RPS-OL compared to GPS.

## VI. System Integration

For integrating all aforementioned counter-drone modules on a single UAS agent, both hardware and software solutions are employed. This section analyses the hardware and software aspects of the integration that provide the capability of having an autonomous counter-drone agent. This solution can still receive telemetry data from the agent and transmit any commands to the agent in case there is a need to utilize the Robot Operating System (ROS) [29].

### A. Hardware Integration

In order to combine all pre-aforementioned mechanisms into a single automated system, there is a need for a processing unit that would be physically small enough to be embedded on the agent but yet computationally powerful enough to cover all needs. At the same time, the processing unit needs to be compatible with all mechanisms. The NVIDIA Jetson Nano Developer Kit was chosen board, as it can be used to implement and integrate all mechanisms of an autonomous, intelligent agent. This agent can detect, track, and counter a malicious UAV while being able to perform self-localization in order to ensure its safe return. The Jeston Nano is equipped with an 128-core Maxwell graphics processing unit that supports NVIDIA's CUDA hardware acceleration. Moreover, the processing unit is also equipped with a quad-core ARM A57 central processing unit and four Gigabytes of LPDDR4 RAM.

Along with the onboard processing unit, the system needs the appropriate hardware that will enable the communication between the onboard processing unit and the UAS. That will enable the control of the UAS and the reception of the telemetry data, and most importantly the camera's feed. A power supply unit, which consists of a DC to DC buck converter, lowers the power output of the UAS to the required 5 volts, 4 amperes DC for the Jetson Nano. Furthermore, a WiFi enabling module that ROS utilizes to send and receive data to and from the system's observer is installed. Finally, an SDR that is used for the self-localization and the jamming system, is mounted at the bottom of the UAS with a custom 3D printed bracket. Figure 9 illustrates all the hardware attached on a custom designed and 3D printed bracket mounted on the top of the UAS, without abstracting either the view of the cameras, nor the GPS and communication signals of the UAS. Further, Fig. 10 shows a basic block diagram of how all counter-drone hardware modules are interconnected.

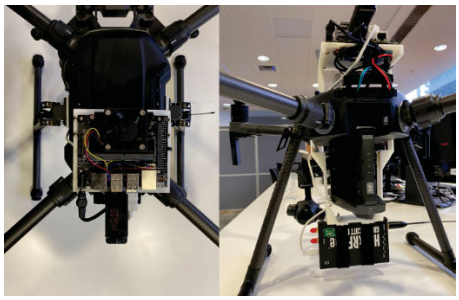


Fig. 9: Picture of the UAS platform with all hardware equipment attached.

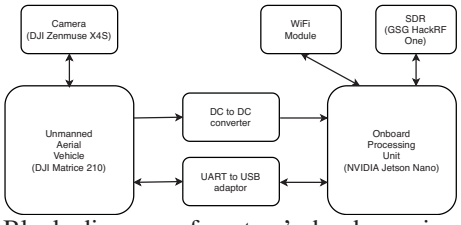


Fig. 10: Block diagram of system's hardware implementation.

### B. Software Integration

The onboard processing unit runs Ubuntu 18.04 LTS which is the operating system that is compatible with all required libraries and programming languages. Furthermore, Ubuntu's compatibility with ROS enables the observer's communication with the system. OpenCV is also compiled along with NVIDIA's CUDA which provides a hardware acceleration for the detector and the tracker. The combination of OpenCV and CUDA alongside Darknet V3 is able to detect potentially malicious UASs at a rate of at least 15FPS. This is a promising result for an edge computing system running on a high resolution camera feed. Moreover, the required software of the SDR is installed allowing the capture of signals of opportunity for the self-localization and the signal transmission for the jamming implementation. Finally, the UAS's manufactures' SDK is employed on the processing unit, thus giving the ability to collect various data from the drone, including the camera stream, as well as instructing the flight controller. The data are then utilized to feed the detection, track, and self-localization algorithms. Hence, the system can autonomously control the UAS, resulting in a self-contained, autonomous, intelligent system.

### C. System Limitations

A number of tracking and interception experiments were carried out to validate the proposed system in real-world settings. From those experiments important insights were gained for future improvements. For instance, using the prescribed hardware, a maximum image processing capacity of 30fps at 720p resolution was achieved that significantly limited our tracking peformance especially at higher travelling speeds of the drones. limits the performance of the detector and tracking component. Moreover, the jammer component introduces a limitation to the counter-attack range (20m Line of sight-LoS) of the anti-drone system. Ofcourse, this range is heavily affected by the environmental conditions but also by the maximum transmit power which had to be kept low due to the weight limitations of introducing larger power amplifiers and power supplies.

## VII. Conclusion and Future Work

In this work, an innovative and low-cost autonomous anti-drone system is proposed, that consists of a complete detection, tracking, jamming, and relative localization system, that can be utilized as an efficient solution for countering rogue drone operations. Each component's performance was evaluated and validated through a number of experiments, and all software and hardware modules were implemented and tested on a commercially available UAS platform (as indicated in www.horizon-block.com). The proposed system was evaluated and demonstrated in a number of outdoor experiments.

Ongoing work includes improvements of the jamming component utilizing different techniques that can allow jamming of multiple targets. In addition, improvements on detection and tracking components include the expansion of the dataset for training purposes so as to improve performance.

## Acknowledgment

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551 (KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

## References

[1] "Drone industry analysis - Research on the growth, size and future — CompTIA", 2020. [Online]. Available: www.comptia.org/content/research/drone-industry-trends-analysis.

[2] I. Guvenc, et al., "Detection, tracking, and interdiction for amateur drones," *IEEE Communications Magazine*, 56(4):75–81, 2018.

[3] J. Loeb, "Exclusive: Anti-drone technology to be tested in UK amid terror fears," *Engineering Technology*, 12(3):9–9, 2017.

[4] BBC, "Eagles trained to take down drones," 2016. [Online]. Available: https://www.bbc.com/news/av/worldeurope-35750816/eagles-trained-to-take-down-drones

[5] K. Wesson and T. Humphreys, "Hacking drones," *Scientific American*, 309(5):54–59, 2013.

[6] H. M. Arthur, "Counter-drone systems," *Center for the Study of the Drone*, Bard College, 2018.

[7] X. Shi, et al., "Anti-UAS system with multiple surveillance technologies: Architecture, implementation, and challenges," *IEEE Communications Magazine*, 56(4):68-74, 2018.

[8] T. Multerer, et al., "Low-cost jamming system against small drones using a 3D MIMO radar based tracking," *Proc. European Radar Conference (EURAD)*, 2017.

[9] J. Rothe, et al., "A concept for catching drones with a net carried by cooperative UAVs," *Proc. IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2019.

[10] O. Westerlund and R. Asif, "Drone hacking with raspberry-pi 3 and WiFi pineapple: Security and privacy threats for the Internet-of-Things," *Proc. 1st Intern. Conf. on Unmanned Vehicle Systems (UVS)*, 2019.

[11] W. Chen, et al., "Manipulating drone dynamic state estimation to compromise navigation," *Proc. IEEE Conference on Communications and Network Security (CNS)*, 2018.

[12] M. R. Brust, et al., "Defending against intrusion of malicious UAVs with networked UAV defense swarms," *Proc. IEEE 42nd Conference on Local Computer Networks Workshops*, 2017.

[13] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", arxiv, abs/1804.02767.

[14] Z. Yang, et al., "Parallel image processing based on CUDA," *2008 International Conference on Computer Science and Software Engineering*, 3:198–201, 2008.

[15] A. Kaehler and G. Bradski, *Learning Opencv 3: Computer Vision In C++ With The Opencv Library*, O'Reilly Media, 2016.

[16] M. Maaref and Z. M. Kassas, "Ground vehicle navigation in GNSS-challenged environments using signals of opportunity and a closed-loop map-matching approach," *IEEE Transactions on Intelligent Transportation Systems*, pp.1–16, DOI: 10.1109/TITS.2019.2907851, June 2019.

[17] H. Simkovits, et al., "Navigation by inertial device and signals of opportunity," *Signal Processing*, 131:280–287, 2017.

[18] K. Shamaei, et al., "Exploiting LTE signals for navigation: Theory to implementation," *IEEE Transactions on Wireless Communications*, 17(4):2173–2189, 2018.

[19] Z.Z.M. Kassas, et al., "I hear therefore I know where I am: Compensating for GNSS limitations with cellular signals," *IEEE Signal Processing Magazine*, 34(5):111–124, 2017.

[20] A.H. Michel, "Counter-drone systems," *Center for the Study of the Drone Technical Report*, 2018.

[21] R. Kapoor, et al., "UAV navigation using signals of opportunity in urban environments: A review," *Energy Procedia*, 110:377–383, 2017.

[22] A.J. Cooper, et al., "A dynamic navigation model for unmanned aircraft systems and an application to autonomous front-on environmental sensing and photography using low-cost sensor systems," *Sensors*, 15(9):21537–21553, 2015.

[23] J.J. Morales and Z.M. Kassas, "Distributed signals of opportunity aided inertial navigation with intermittent communication," *Proc. 30th International Technical Meeting of the Satellite Division of the Institute of Navigation*, 2018.

[24] J. Morales and Z. M. Kassas, "Event-based communication strategy for collaborative navigation with signals of opportunity", *Proc. 52nd Asilomar Conference on Signals, Systems, and Computers (ACSSC)* 2018.

[25] M. Maaref, et al., "Lane-level localization and mapping in GNSS-challenged environments by fusing lidar data and cellular pseudoranges," *IEEE Transactions on Intelligent Vehicles*, 4(1):73–89, 2018.

[26] K. Shamaei and Z. M. Kassas, "LTE receiver design and multipath analysis for navigation in urban environments," *Navigation*, 65(4):655–675, 2018.

[27] J.F. Raquet, et al., "Issues and approaches for navigation using signals of opportunity," *Proc. Institute of Navigation National Technical Meeting*, 2:1073–1080, 2007.

[28] J.F. Raquet, "Navigation using pseudolites, beacons, and signals of opportunity", *NATO STO SET-197, Navigation Sensors and Systems in GNSS Degraded and Denied Environments*, pp. 1-18, 2013.

[29] Quigley, Morgan, et al. "ROS: An open-source robot operating system", *ICRA Workshop on Open Source Software*, 2009.

[30] H. Silva, "Experimental study on RSS based indoor positioning algorithms," *Transactions on Engineering Technologies*, Springer, pp. 451–466, 2016.

[31] S. Tomic, et al., "On target localization using combined RSS and AoA measurements," *Sensors*, 18(4):1–25, 2018.

[32] W. Powell and I. Ryzhov, *Optimal learning*, 2nd ed, Wiley, 2012.

[33] I. Ryzhov, et al., "The knowledge gradient algorithm for a general class of online learning problems," *Operations Research*, 60(1):180–195, 2012.

[34] W. Powell, "The knowledge gradient for optimal learning," *Wiley Encyclopedia of Operations Research and Management Science*, 2010.