A distributed UAV analytics framework for DAO-based swarm systems

A. Vasalos[1], A. Economopoulos[1], A. Oikonomakis[1], A. Chakraborty[2], M.-A. Kourtis[1], G. Alexandridis[3], W. Tavernier[2], G. Xilouris[1], I. Chochliouros[4], I. Vasalos[1], and P. Trakadas[3]
[1]National Center for Scientific Research "Demokritos", Greece
[2]IDLab, Ghent University–imec, Ghent, Belgium
[3]National and Kapodistrian University of Athens, Athens, Greece
[4]Hellenic Telecommunications Organization S.A. (OTE), Athens, Greece

Abstract— **Unmanned Aerial Vehicles (UAVs) are increasingly deployed in inspection and monitoring missions, yet onboard computation and communication impose significant energy burdens that limit flight time and operational scope. In this work, we introduce a novel, blockchain-enabled framework—grounded in the Distributed Autonomous Organization (DAO) paradigm—for orchestrating distributed analytics across a swarm of UAVs. Leveraging the OASEES project's smart-contract architecture, each drone embeds a Metrics Module for real-time power monitoring, a Behavioral Module for adaptive control, and a Blockchain Agent that autonomously proposes, votes on, and executes collective decisions. Three concurrent threads—Proposal Trigger, Voting, and Action Execution—enable fully decentralized governance of swarm behavior: from detecting critical energy thresholds and formulating swarm-wide conservation maneuvers, to executing approved strategies across all members. We validate our framework in a UAV-based infrastructure inspection scenario, employing a YOLOv5 object-detection pipeline to classify four corrosion classes on a telecommunications mast under three video-capture modalities (short-distance, long-distance, and horizontally concatenated streams). Across all configurations, our system achieves near-perfect precision, recall, and mean Average Precision (mAP50 − 95 ≈ 0.995), demonstrating both the efficacy of distributed workload inference and the feasibility of treating a single drone as a multi-feed processor. These results underscore the potential of DAO-driven UAV swarms for energy-aware, resilient aerial analytics, and pave the way for fully decentralized 5G/6G-enabled airborne networks.**

Keywords— Decentralized Data Processing, Drone Swarm, Edge Computing

I. Introduction

The emergence of Unmanned Aerial Vehicles (UAVs) in various technological domains and sectors as enablers, has created different opportunities and needs for the deployment of energy efficient and cost-effective devices. UAVs have evolved enough to be able to process workloads and perform computations while on operation and on-air, but with the trade-off of increased energy consumption. Energy is a critical aspect in UAV operation as it directly affects flight time and duration of the mission, thus every component or design decision that saves energy and battery life can greatly benefit UAV operation.

An alternative approach on optimized UAV operation is the distribution of workloads among different UAVs that can comprise a swarm. Swarm architectures and logic can improve UAV operation efficiency and generate new opportunities in an emerging market [1]. [2] provides insight into ethical aspects and the use cases of UAV swarms in various military, civilian, and entertainment applications.

Studies by Azari and al. [3] explore the deployment of terahertz communication for UAV regarding 6G communication, sensing localization and channel modelling. Finally Jiang et al. [4] offer a comprehensive survey on energy-efficient UAV communication in 6G, focusing on energy models, designs, and open issues [5].

In this paper, a swarm-based approach for UAV distributed processing is presented based on the OASEES HE project. OASEES proposes a blockchain based approach on swarm intelligence and how this can benefit UAV infrastructure inspection scenarios leveraging distributed workload inference among different swarm UAV members. This approach is based on the Distributed Autonomous Organization (DAO) paradigm [7], where a group of devices can collectively vote on a proposal and take a decision as a group. This architectural concept is applied in an actual deployment of UAV infrastructure inspection, using object detection, for single and collective AI workload processing. The paper is organized as follows: Section 2 provides the DAO based approach architecture and workflows regarding the UAV use case. Section 3 provides the validation and evaluation of the object detection service for single UAV and swarm-based processing. And finally, section 4 concludes the paper and its results.

II. Swarm – DAO Interaction

A. Drone Software Architecture

The operational autonomy and decentralized coordination of each drone within the swarm is enabled by the underlying software architecture. With an emphasis on segmenting the software's functionality into data collection, behavior execution, and blockchain interaction, our architecture is made to be modular and is comprised of three core components: the Metrics Module, the Behavioral Module, and the Blockchain Agent. Figure 1 provides a high-level overview of their interconnectedness.

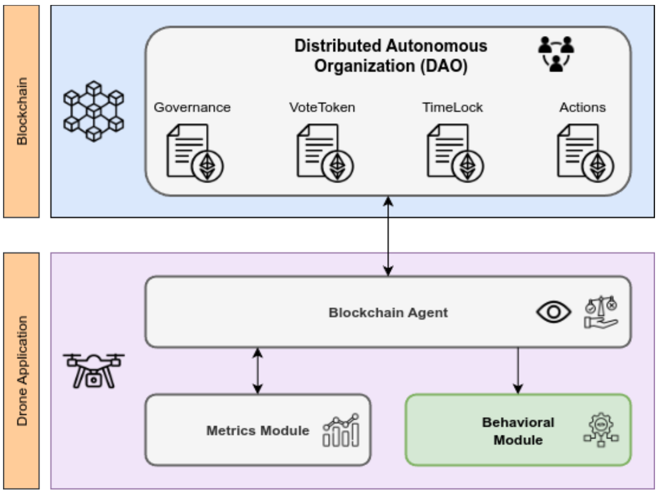

Figure 1: Overview of drone software architecture

The **Metrics Module** acts as the main input for the drone's decision-making mechanism. Its functionality is to collect essential operational data, both individual and neighborhood-wide, and subsequently make them available via a standardized endpoint. By making these metrics programmatically available, the module offers the raw data necessary to enable individual, as well as collective intelligence, i.e. the drone can assess its own state and, by extension, contribute to an aggregated view of the swarm's status. For our application, the main measure is Power Consumption, as it can offer a direct and real-time reflection of both the drone's own sustainability as well as the local cluster's energy status

The **Behavioral Module** encapsulates the drone's application logic. It is implemented as a collection of discrete actions or states that define the drone's operational capabilities — for instance, “Break Off from the Swarm” “Reunite with the Swarm” — where each behavior is mapped to a specific endpoint, allowing the drone's operational mode to be altered dynamically through API calls. Instead of containing the complex decision-making logic itself, the Behavioral Module can be perceived as an execution engine, effectively differentiating the "how" of an action from the "why" that is decided through the DAO.

The **Blockchain Agent** is the central part of the architecture, integrating the Metrics and Behavioral modules and managing all communication with the DAO's smart contracts (i.e. Governance, VoteToken, TimeLock, and Actions). It is the component that enhances the drone’s isolated operation with "DAO-aware" capabilities, enabling it to participate in the governance of the swarm. The agent’s functionality can also be broken down to the concurrent execution of three monitoring threads:

Proposal Trigger Thread: This thread continuously polls the Metrics Module's endpoint to monitor the drone's individual / neighborhood metrics. Upon receiving them, predefined rules are applied and the decision-making process is kickstarted—for instance, if the average power consumption of the drone and its neighbors rises above a critical threshold, the agent concludes that a change in swarm strategy is necessary. It then formulates and submits a proposal to the DAO's Governance contract, suggesting a new behavior (e.g., a swarm-wide "Conserve Energy" state).

Voting Thread: This thread monitors the Governance contract for any new or active proposals submitted by other members of the DAO. When an active proposal is detected, the drone can reconsult its Metrics Module as a means to assess its current state (relative to the proposal’s objective) and make an informed decision. For instance, if a proposal suggests a high-energy surveillance maneuver, the drone will check its own power levels. Based on this local context, it casts its vote for or against the proposal, contributing its own assessment to the collective consensus.

Action Execution Thread: This thread monitors the state of a dedicated DAO smart contract which reflects the swarm’s current status. Whenever a proposal is successfully passed and executed by the DAO, a change in the aforementioned smart contract’s state directly implies a specific change in swarm behavior. The Action Execution Thread detects this change, interprets the new state value, and makes the corresponding API call to the drone's Behavioral Module. This sequence triggers a new individual behavior for each of the drones, eventually leading to a shift in swarm-wide operations. This closes the loop, translating a decentralized decision into a coordinated, real-world action.

B. Execution Sequence in our System

In order to visualize the end-to-end functionality of our software architecture, as well as map its components to our proposed system, this subsection presents a series of sequence diagrams. These diagrams illustrate the complete lifecycle of a decentralized decision, from the initial data- driven trigger within a single drone to the synchronized adaptation of the entire swarm's behavior.

The diagram in Figure 2 details the operation of the Proposal Trigger Thread. The Blockchain Agent periodically polls the Metrics Module and parses the response. Depending on whether the returned data (i.e. power consumption in our case) has crossed a predefined threshold, the agent initiates a transaction to the Governance smart contract, formally creating a new proposal to alter the swarm's behavior.

Figure 3’s diagram illustrates the Voting Thread's workflow. Its core application flow is initiated when at least one “Active” Proposal is detected. To make an informed decision, it again queries its local Metrics Module for a real-time assessment of its own power status. Based on this data, it casts its vote.

The next diagram (Figure 4) is a depiction of the final stages of the DAO's governance process for a successful proposal. It shows the calls to queue and subsequently execute the proposal via the TimeLock contract, which ultimately modifies the state of the "Actions" smart contract (dedicated to represent the Swarm’s status). Crucially, this diagram also introduces our system’s Human-in-the-Loop (HITL) intervention capability. This mechanism provides the option for a human operator to directly interact with the system, providing a critical layer of oversight and ensuring the robustness and safety of the swarm's operation by having the ability to override or trigger actions immediately.

The final diagram (Figure 5) visualizes the Action Execution Thread, which closes the operational loop. The Blockchain Agent constantly monitors the "Actions" smart contract for state changes. If the value remains the same, no action is taken. However, upon identifying a new value—the result of a successfully executed proposal—the agent interprets it and makes the corresponding API call to its Behavioral Module. This results in the either breaking off from the cluster to process the rest of the swarm’s feeds, sending the feed to the newly elected “Processing” Drone, or reuniting with the rest of the swarm.

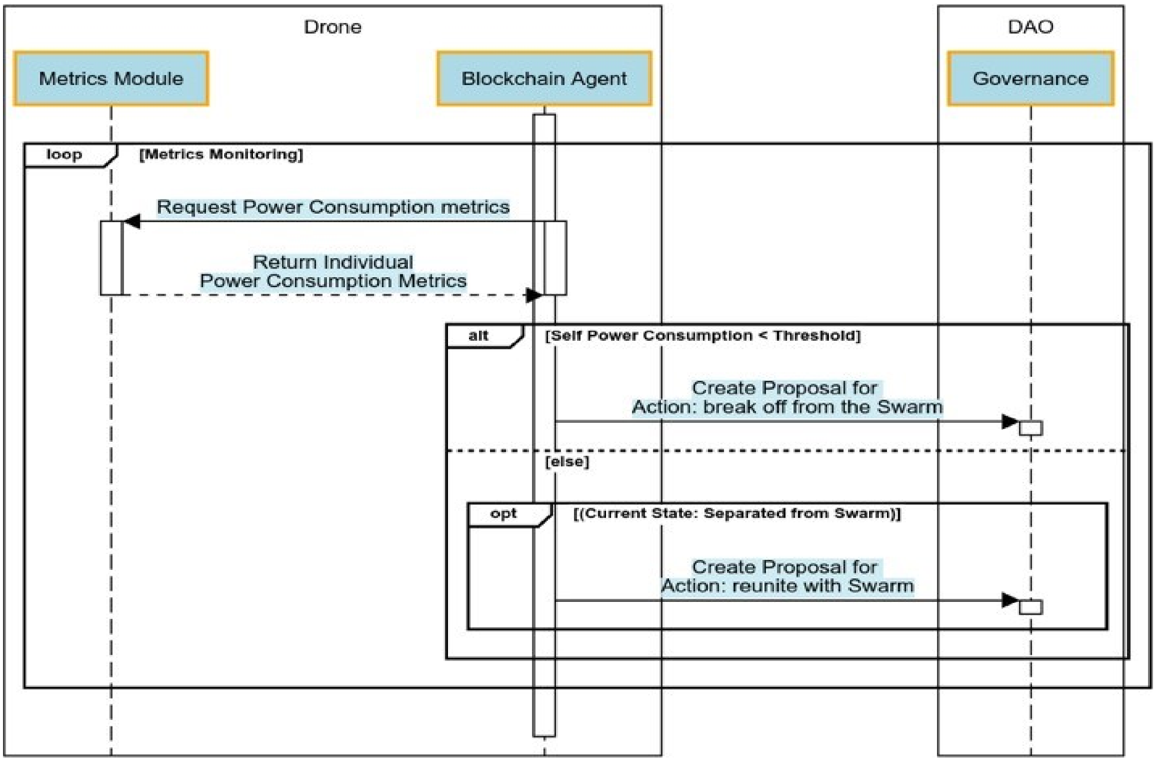

Figure 2: Metrics Monitoring and Proposal Creation

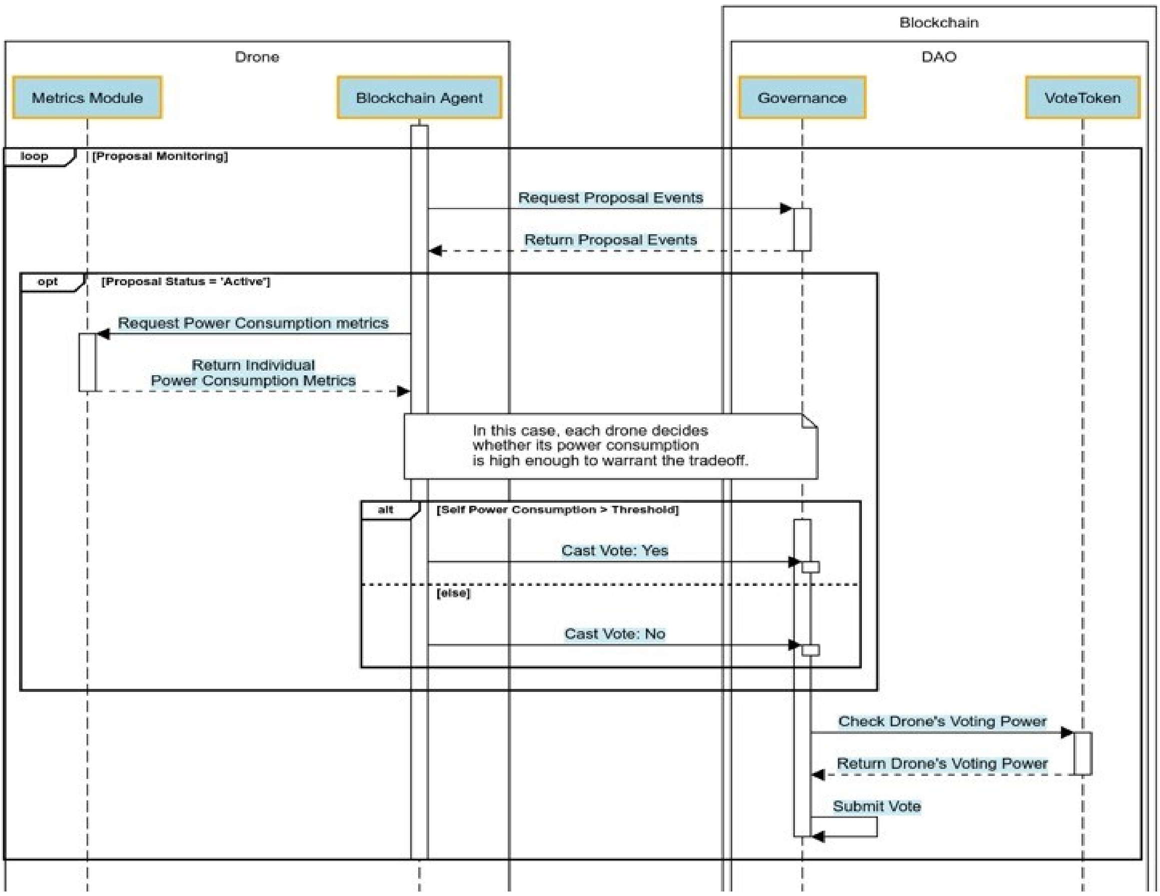

Figure 3: Drone - DAO Voting Process

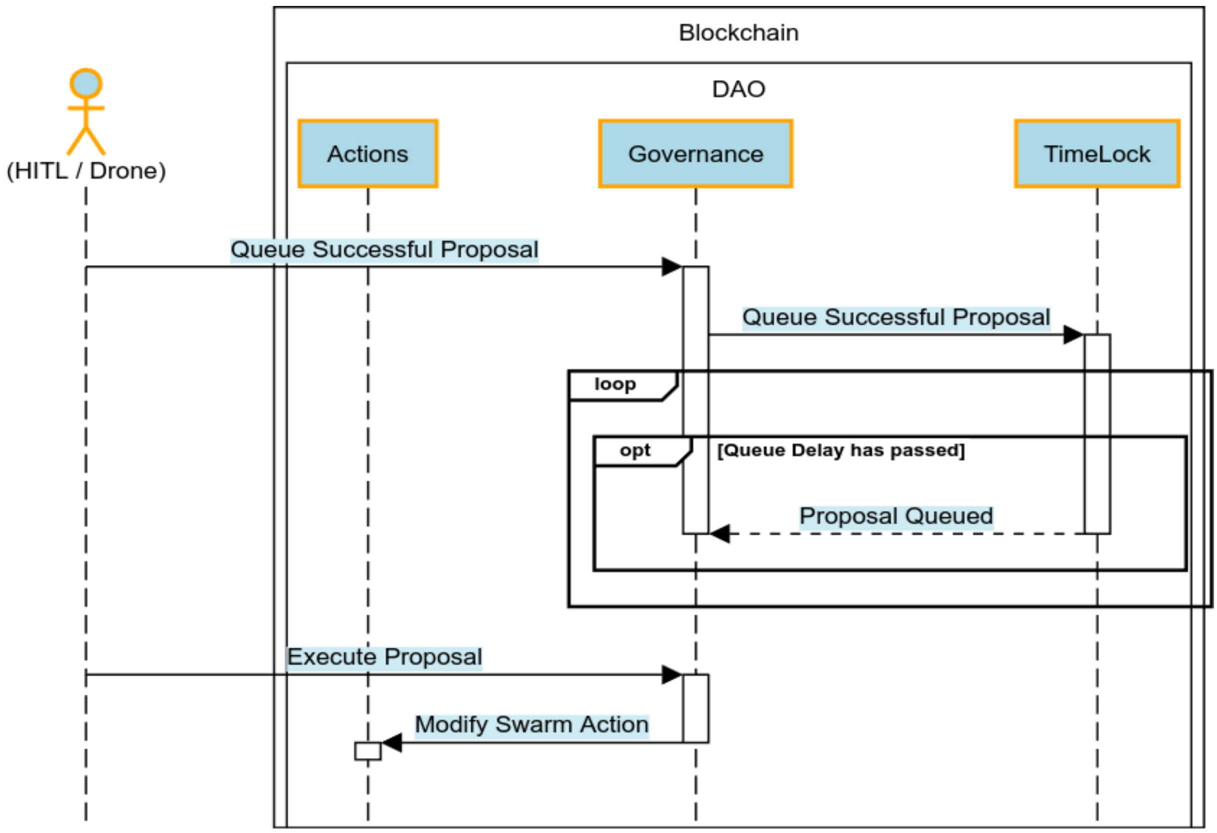

Figure 4: Successful Proposal Execution

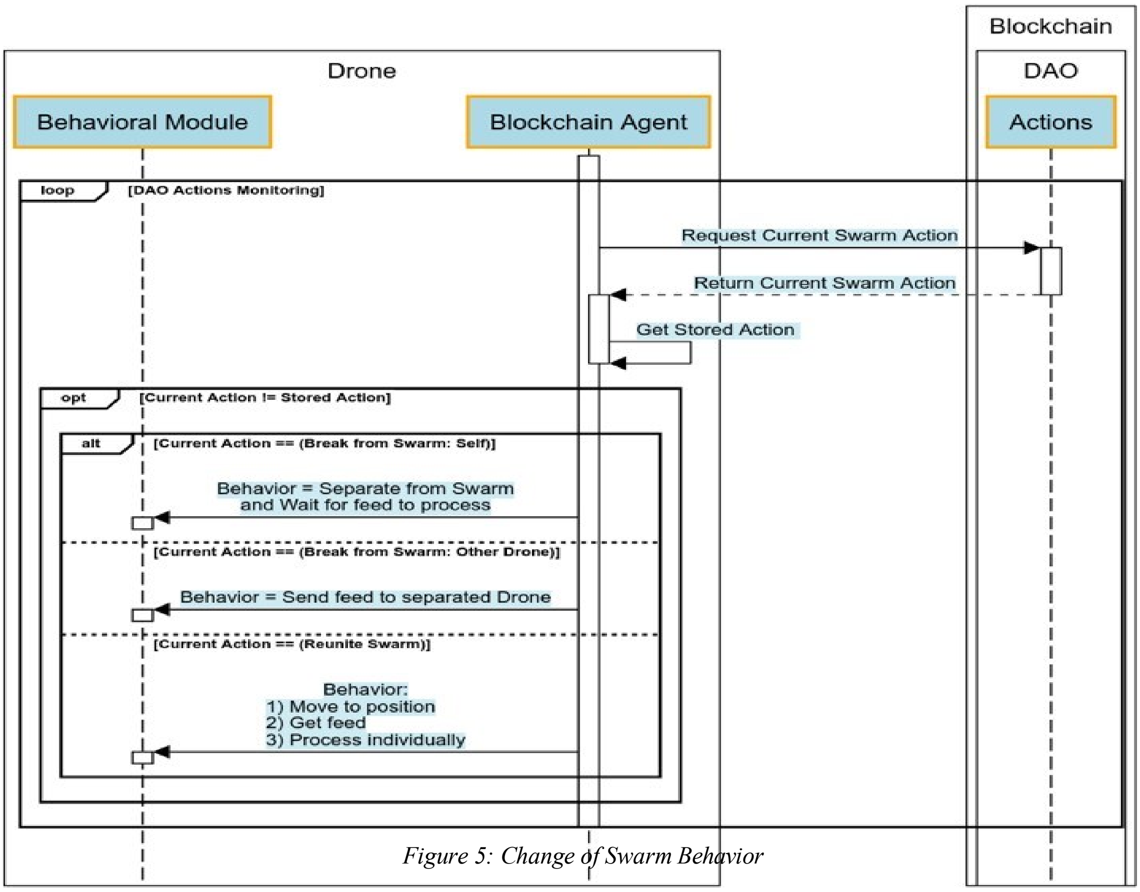

Figure 5: Change of Swarm Behavior

III. Detection and Classification

Object detection involves two main issues: classification of an object i.e detection and recognition of the object and localization of the object i.e. discovery of the location of the object in the image by means of bounding boxes. Most object detection methods utilize a Convolutional Neural Network (CNN). A CNN uses learnable convolutional layers i.e. trainable filters or kernels to scan for patterns and learn the spatial hierarchies of features in an image, and processes images using a grid topology. In this way CNNs are responsible for learning to detect objects and predict bounding boxes efficiently. Core operations in a CNN are: Convolution i.e. feature or object extraction and detection of patterns, Batch Normalization i.e. normalization of layer inputs after every convolutional layer, Activation i.e. introduction of non linearity into the network to model complex relationships, Pooling i.e. reduction of spatial resolution or downsampling to retain dominant features and focus on them, Upsampling i.e. restoration of feature size and spatial dimensions for detection at multiple scales.

Convolutional Layer

A convolutional layer performs a sliding dot product between a kernel (filter) and local patches of the input feature map i.e. an image.

Mathematical Operation:

$$\boldsymbol{Y(i,j) = \sum_{m=1}^{M}\sum_{n=1}^{N} K(m,n)\cdot X(i+m,j+n)}$$

Where:

- X = input
- K = kernel
- Y = output feature map

These kernels learn to detect edges, textures, parts of objects, and entire objects

One of the algorithms in which CNNs are employed is YOLOv5 [5]. YOLOv5's convolutional network is divided into three main components: Backbone, Neck and Head. All the aforementioned CNN operations are performed in Yolov5 algorithm. Input downsampling in Focus layer with minimal information loss is performed in Backbone and Head. Convolution with varying kernel sizes to extract spatial patterns and objects from images is performed in all layers. Batch Normalization follows every convolutional layer. Sigmoid-weighted Linear Unit (SiLU) activation which introduces non-linearity is used in most convolutional blocks for smoother gradient flow and better convergence. C3 Module or CSP Bottleneck which performs deep feature extraction is used since it contains residual flow or residual connections and allows better gradient flow, so it improves efficiency. Residual flow refers to the use of residual connections where a layer's input is added directly to its output. C3 Module is performed in Backbone and Head. Spatial Pyramid Pooling – Fast captures multi-scale context by pooling at different kernel sizes in order to increase the receptive field i.e. region of image and focus on dominant features. Spatial Pyramid Pooling – Fast is performed in Backbone. Upsampling is performed in the Neck component. The Head component of the YOLOv5 architecture is the final stage in the model, it takes the already processed feature maps from the Neck and performs object detection predictions. It applies lightweight convolutional operations to produce predicted bounding boxes coordinates, scores counting the number of times an object is detected and class probabilities.

In order to validate the predicted data, in Yolov5 the predicted bounding boxes coordinates are compared against the ground truth annotation data that consists of bounding boxes coordinates and class labels for each box. Hence object detection metrics are produced so as to evaluate the algorithm performance. These metrics consist of : Precision (P) which is the proportion of true positive detections among all positive detections. Recall (R) which is the proportion of true positives among all ground truth objects. mAP50 which is the mean Average Precision(mAP) at Intersection over Union (IoU)=0.5. mAP0.5-0.95 which is the mean Average Precision(mAP) averaged over multiple IoU thresholds from 0.5 to 0.95 in steps of 0.05.

Considering the use case addressed in this paper, the goal is to detect and classify corrosion on a metallic telecommunications mast. Hence four classes of corrosion have been considered: corrosion, moderate corrosion, rust and severe corrosion. In order to evaluate the algorithm performance, corrosion classification is performed in 3 video streams. The first video stream is produced at a shorter distance from the mast. The second video stream is produced at a longer distance from the mast. In the third case the two videos are horizontally concatenated. For all streams predicted bounding boxes, scores or instances counting the number of times a class occurs, class probabilities and metrics are produced.

Figure 6 shows images with predicted bounding boxes and class probabilities generated from the video that was produced at a shorter distance from the mast. Table 1 shows the number of class instances and metrics.

Figure 6: shorter distance generated images with predicted bounding boxes and class probabilities

Class	Images	Instances	P	R	mAP 50	mAP 50-95
all	9	81	1	1	0.995	0.995
corrosion	9	8	1	1	0.995	0.995
moderate corrosion	9	51	1	1	0.995	0.995
rust	9	14	1	1	0.995	0.995
severe corrosion	9	8	1	1	0.995	0.995

Table 1: Class instances and metrics for shorter distance generated images.

Figure 7 shows images with predicted bounding boxes and class probabilities generated from the video that was produced at a longer distance from the mast. Table 2 shows the number of class instances and metrics.

Figure 7: longer distance generated images with predicted bounding boxes and class probabilities.

Class	Images	Instances	P	R	mAP50	mAP50-95
all	9	49	1	1	0.995	0.995
corrosion	9	4	1	1	0.995	0.995
moderate corrosion	9	9	1	1	0.995	0.995
rust	9	15	1	1	0.995	0.995
severe corrosion	9	21	1	1	0.995	0.995

Table 2: Class instances and metrics for longer distance generated images.

Figure 8 shows images with predicted bounding boxes and class probabilities generated from the combined i.e. horizontally concatenated videos. Table 3 shows the number of class instances and metrics.

Figure 8: generated images from combined videos with predicted bounding boxes and class probabilities.

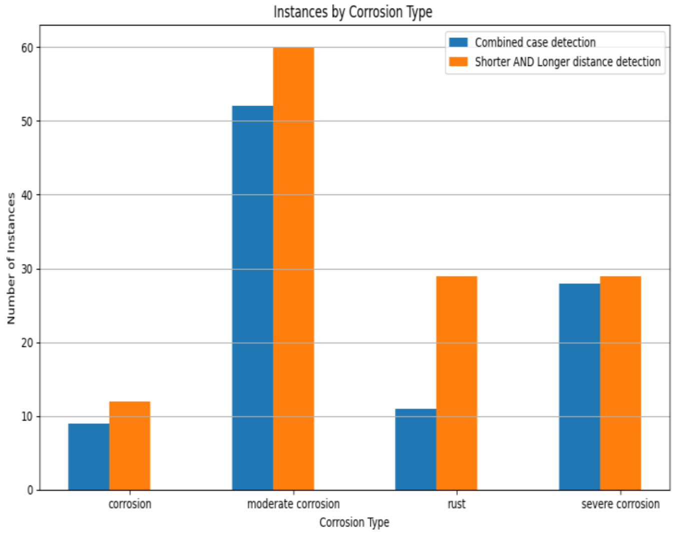

Figure 9: the detected instances of the generated images from the combined i.e. horizontally concatenated videos are plotted against the sum of detected instances from the longer and shorter distance generated images.

Class	Images	Instances	P	R	mAP 50	mAP 50-95
all	9	100	1	1	0.995	0.995
corrosion	9	9	1	1	0.995	0.995
moderate corrosion	9	52	1	1	0.995	0.995
rust	9	11	1	1	0.995	0.995
severe corrosion	9	28	1	1	0.995	0.995

Table 3: Class instances and metrics for generated images from combined videos.

Figure 9 shows the total number of class instances that are detected. Hence the detected instances of the generated images from the combined i.e. horizontally concatenated videos are plotted against the sum of detected instances from the longer and shorter distance generated images.

IV. Conclusion

These findings affirm the utility of the OASEES framework in managing the operational demands of 5G/6G-enabled UAVs in various situations. The proposed architecture aligns with the trend towards network decentralization, empowering end-users and enhancing the efficiency and resilience of connectivity. The concept of a UAV acting as a flying 5G base station underlines a fresh perspective in network connectivity solutions. It presents not only a technical innovation but also an intriguing business opportunity. Our research underscores the immense potential of this approach, thereby marking an important milestone in the journey towards universal Internet access.
Based on the outlook of UAV-enabled connectivity solutions, it is clear that the OASEES framework will play a vital role. The potential of the technology and the opportunities it presents warrant further exploration and development. Future work in this area will delve deeper into the nuances of the proposed architecture and seek ways to improve and optimize it for a variety of applications and scenarios.

Considering the use case addressed in this paper, predicted bounding boxes using YOLOv5 algorithm were compared to the ground truth annotation data. From the comparison and according to the metrics : P (precision), R(recall) and mean precision mAP, mAP50-95 a very good match was obtained between predictions and ground truth data since all these metrics were high. Therefore corrosion detection and classification on a metallic telecommunications mast is feasible and prediction instances are considered trustworthy. Additionally an effort was pursued to apply detection algorithms on horizontally concatenated videos. The number of detected instances of the generated images from the horizontally concatenated videos are comparable and similar to the sum of detected instances from the longer and shorter distance generated images. Difference is due to the fact that when concatenating videos the dimensions of the videos change so detection algorithms are expected to yield different numbers of prediction instances. However it is demonstrated that object detection on concatenated videos is feasible so the detection algorithm can be applied on one drone which receives the broadcasted videos from all the other drones of the swarm.

Acknowledgment

The research work presented in this article has been supported by the European Commission under the Horizon Europe Programme and the OASEES project (no. 101092702).

References

[1] Zhou, Yongkun, Bin Rao, and Wei Wang. "UAV swarm intelligence: Recent advances and future trends." Ieee Access 8 (2020): 183856-183878.

[2] S. Javed et al., "State-of-the-Art and Future Research Challenges in UAV Swarms," in IEEE Internet of Things Journal, vol. 11, no. 11, pp. 19023-19045, 1 June1, 2024, doi: 10.1109/JIOT.2024.3364230.

[3] M. M. Azari, S. Solanki, S. Chatzinotas, and M. Bennis, "THz-Empowered UAVs in 6G: Opportunities, Challenges, and Trade-Offs," IEEE Communications Magazine 60(5):24-30 May 2022 DOI:10.1109/MCOM.001.2100889

[4] X. Jiang et al., "Green UAV communications for 6G: A survey," Chinese Journal of Aeronautics, vol. 35, no. 9, pp. 19–34, Sept. 2022.

[5] Nazir, Aabidah & Wani, Mohd. Arif. (2023). You Only Look Once - Object Detection Models: A Review. Proceedings of the 17th INDIACom – 10th International Conference on Computing for Sustainable Global Development (INDIACom 2023), pp. 1088–1095

[6] Q. Wang et al., "A First Look into Blockchain DAOs," 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates, 2023, pp. 1-3, doi: 10.1109/ICBC56567.2023.10174961.