Autonomous Based Visual Navigation Engine for Drones

Malika Hafiza Pasha, *Student, CSUDH*

***Abstract*—The main motivation of the paper is to make the drone autonomous and make to travel anywhere without any human support. Drones plays an important role in specific situations or the scenarios. Existing Drones have a problem of battery issues, and they are uncapable for travelling for the long distances. These problems can be easily covered by using the deep neural network. We are implementing this using the GAP architecture and with help of deep neural network in the cloud we can achieve this process. To deploy the visual navigation algorithm, we can design a lightweight, modular, and configurable PCB with the highly optimized layout and a form factor compatible with the nano-size quadrotor. It also has the PULP based GAP8 SOC which has two cypress hyper bus memories and an ultralow-power himax CMOS image sensor to capture the images and videos. With these features we can design a drone that consume the low power and it also has the feature of autonomous system. With the help of the deep neural network, it has the capability to train itself and it can navigate itself. Drones can be useful in many ways for example if there is any natural disasters drones can go into the place where human cannot go and it can be used as surveillance camera and give report about causalities happened in the disaster. Autonomous can also be used by the army to locate and trace the criminal activities. The use of the GAP architecture and the deep neural network makes the engine flexible and can be used to span a wide performance range as its peak performance corner.**

I. Introduction

THE Drones play a significant role across various applications, particularly in the context of the growing Internet of Things (IoT) era and the advancement of artificial intelligence in embedded systems. Traditional isolated systems are progressively being replaced by AI-based sensor nodes capable of independently acquiring, processing, understanding information, and interacting with each other. The ideal IoT node should possess the capability to autonomously navigate its environment while sensing, analyzing, and comprehending its surroundings. Given their aerial nature, drones prove versatile in different environments, showcasing speed, agility, and the ability to rapidly gather information from both onboard sensors and devices in the surroundings. Additionally, they can conduct advanced onboard analytics, filtering essential information before transmitting it to centralized servers, typically cloud-based, for storage and further processing. Drones are well-suited for indoor applications where safe operation is crucial, as well as in densely populated urban areas, showcasing both sense-and-act capabilities. interact with the surroundings. This Scenario can also be used as the application for intelligent Nano-sized UAVs can potentially serve for online wireless activity detection through the deployment of edge nodes in the environment equipped with onboard radio packet sniffing. Commercial off-the-shelf (COTS) quadrotors are now entering the nanoscale, featuring dimensions of a few centimeters in diameter, and weighing only a few tens of grams. However, these commercial nano-UAVs still lack the autonomy seen in their larger counterparts due to computational limitations imposed by their tiny power envelopes, rendering them inadequate for sophisticated AI workloads.

The conventional approach to autonomous UAV navigation involves a localization-mapping-planning cycle. This cycle includes estimating robot motion through off-board methods (e.g., GPS) or onboard sensors (e.g., visual-inertial sensors), constructing a local 3-D map of the environment, and planning a safe trajectory. These methods, though effective, are computationally expensive for platforms with constrained resources. Recent findings have demonstrated that lighter algorithms, relying on convolutional neural networks (CNNs), are sufficient for basic reactive navigation of small drones, even without a pre-existing map of the environment. Unfortunately, their computational and power requirements still surpass the allocated budget for the navigation engines of current nano-drones, which primarily rely on simple, low-power microcontroller units (MCUs).

Wood suggests that, for small-sized UAVs, the maximum power budget for onboard computation is limited to 5%, with the majority being allocated to propellers (86%) and low-level control components (9%). Achieving state-of-the-art navigation capabilities for nano- and pico-sized UAVs hinges on the development of energy-efficient and computationally capable hardware, highly optimized software, and innovative algorithms integrated into a next-generation navigation engine. These challenges and requirements mirror those encountered in deploying high-level computational capabilities on IoT edge nodes and sensors. Furthermore, in the case of miniature flying robots, the challenge is heightened by the strict real-time constraints dictated by the necessity for rapid reactions to prevent collisions with dynamic obstacles.

While conventional UAVs, characterized by power capacities in the several hundred Watts range, have the capability to integrate robust high-end embedded computers such as Qualcomm Snapdragon, Droid, NVIDIA Jetson TX1, and TX2, nano-sized UAVs have historically been limited by the

capabilities of microcontroller devices offering at most a few hundred Mop/s. Consequently, the implementation of CNN-based autonomous vision navigation has been deemed unattainable for this category of drones.

This paper introduces a novel visual navigation engine and a comprehensive methodology for deploying complex convolutional neural networks (CNNs) on commercially available, resource-constrained computational edge nodes, specifically designed for nano-sized flying robots. The study presents what is believed to be the initial implementation of a state-of-the-art fully autonomous vision-based navigation system, employing deep learning on a UAV visual navigation engine that consumes less than 284 mW at peak power (64 mW in the most energy-efficient configuration). This system is fully integrated and operates in closed-loop control within the open-source Crazyflie 2.0 nano-UAV. The visual navigation engine, depicted atop the Crazyflie 2.0. It utilizes the GreenWaves Technologies GAP8 System-on-Chip (SoC), an embedded processor known for its high efficiency and capability to execute complex algorithmic flows on power-constrained devices, such as nano-scale UAVs, by leveraging the emerging parallel ultralow-power (PULP) computing paradigm.

The paper makes several notable contributions, extending the state-of-the-art in nano-scale UAVs and serving as a proof-of-concept for a broader range of AI-based applications in the Internet of Things (IoT) domain. Specifically:

1) The authors present a general methodology for deploying cutting-edge deep learning algorithms on ultralow-power embedded computation nodes, including miniaturized robots.
2) They adapt DroNet, a CNN-based approach for autonomous navigation proposed for standard-sized UAVs, to meet the computational requirements of nano-sized UAVs, incorporating fixed-point computation.
3) The deployment of DroNet occurs on the PULP-Shield, an ultralow-power visual navigation module featuring the GAP8 SoC, an ultralow-power camera, and off-chip Flash/DRAM memory. This shield is designed as a pluggable printed circuit board (PCB) for the 27g Crazyflie 2.0 nano-UAV.
4) The methodology is demonstrated for the DroNet CNN, achieving comparable results in terms of UAV control as the standard-sized baseline within a PULP-Shield power budget of just 64 mW, providing a throughput of 6 frames/s and up to 18 frames/s within 284 mW.
5) The authors validate their methodology through a field-proven demonstration, showcasing a closed-loop fully operational vision-driven autonomous navigation relying solely on onboard resources.

To validate the effectiveness of our methodology, we showcase a fully operational closed-loop demonstrator in the supplementary material. In support of advancing research in this domain, we make the design of the PULP-Shield, along with all the code running on GAP8, datasets, and trained networks, publicly accessible under liberal open-source licenses.
The subsequent sections of this paper are organized as follows. Section II provides a comprehensive overview of the state-of-the-art in both nano-UAVs and low-power IoT. Section III outlines the software/hardware background pertinent to this study. In Section IV, we delve into the details of our CNN mapping methodology, elucidating software tools and optimization techniques. The design of the visual navigation engine is expounded in Section V. Section VI-B presents the experimental evaluation, considering both performance and power consumption, with a comparison to the state-of-the-art and an assessment of the final control accuracy. Finally, Section VII serves as the conclusion for this paper.

II. Related work

The expansion of the Internet of Things (IoT) is driving a shift towards edge computing, enhancing scalability, robustness, and security. While current IoT edge nodes are typically immobile, autonomous nano-sized UAVs exemplify the next generation of IoT end-nodes, characterized by high mobility and a need for an unprecedented level of onboard intelligence. The objective of this paper is to align state-of-the-art visual autonomous navigation with ultralow-power nano-drones, facilitating their deployment for IoT applications. Consequently, this section delves into pertinent research on nano-aircrafts and the integration of deep neural networks (DNNs) on low-power IoT nodes.

The conventional approach to autonomous navigation for nano-drones involves outsourcing computation to a remote, powerful base station. For example, Dunkley et al. Devised a visual-inertial simultaneous localization and mapping (SLAM) algorithm for a 25g nano quadrotor, where all computation was conducted off-board by streaming video and inertial information from the drone to a remote, power-unconstrained laptop. The primary challenges with such solutions include latency, maximum communication distance, reliability issues due to channel noise, and high onboard power consumption due to frequent video streaming.

A few prior works have presented nano-sized flying robots with some degree of autonomous navigation relying on onboard computation. McGuire et al. They developed a 4g stereo-camera and proposed a velocity estimation algorithm capable of running on the microcontroller unit (MCU) onboard a 40g flying robot. While this solution enables the drone to avoid obstacles during flight, it still requires favorable flight conditions (e.g., low-flight speed of 0.3m/s). In another study, an optical-flow-based guidance system was created for a 46g nano-sized UAV. The proposed ego-motion estimation algorithm did not rely on feature tracking, allowing it to run on the onboard MCU. Unfortunately, the autonomous functionality was limited to hovering, and the

specific length.

The conventional approach to autonomous navigation for nano-drones typically involves offloading computation to a remote and powerful base station. For example, Dunkley et al. developed a visual-inertial simultaneous localization and mapping (SLAM) algorithm for a 25g nano quadrotor. In this case, all computation was carried out off-board, streaming video and inertial information from the drone to a remote laptop with ample power resources. However, this approach presents challenges such as latency, limited communication distance, reliability issues due to channel noise, and high onboard power consumption from continuous high-frequency video streaming.

Some prior works have explored nano-size flying robots with onboard computation for autonomous navigation. McGuire et al. developed a 4g stereo camera with a velocity estimation algorithm that could run on the onboard MCU of a 40g flying robot. While this allows obstacle avoidance during flight, it still requires favorable flight conditions, such as a low flight speed of 0.3m/s. In another instance, an optical-flow-based guidance system was designed for a 46g nano-size UAV, with an onboard MCU running an ego-motion estimation algorithm. However, the autonomous functionality was limited to hovering, and the method did not achieve the accuracy of computationally expensive techniques based on feature tracking.

In a separate effort, an application-specific integrated circuit (ASIC) called NAVION was introduced for onboard visual-inertial odometry. Although capable of performing state estimation at a high frame rate with low power consumption, this ASIC alone lacks the full range of functionalities required for complete UAV autonomy. In practical use, additional circuits would still be necessary for complementary onboard computation and interaction with the drone's sensors. It's worth noting that as of now, the NAVION accelerator has not demonstrated the same level of maturity and completeness as discussed in this paper, and it has not been tested on a real-life flying nano-drone.

Commercial off-the-shelf (COTS) nano-size quadrotors, such as the Bitcraze Crazyflie 2.0 or the Walkera QR LadyBug, are equipped with low-power single-core MCUs like the ST Microelectronics STM32F4. Despite substantial efforts in academia and industry, including tools like TensorFlow Lite and ARM Compute Library, there is no established consensus on the optimal deployment of complex AI-powered algorithms, such as Deep Neural Networks (DNNs), on these low-power microcontrollers. This challenge involves resource management, particularly in terms of available working memory and storage, as well as the peak throughput achievable by single-core MCUs. The difficulty is exacerbated by the absence of abstraction layers and computing facilities commonly found in mainstream deep learning tools, such as linear algebra libraries (e.g., BLAS, CUBLAS, and CUDNN) and preprocessing libraries (e.g., OpenCV).

ARM's recent release, CMSIS-NN, aims to bridge this gap by accelerating deep inference compute kernels on Cortex-M microcontroller platforms, providing functionalities similar to a BLAS/CUDNN library. However, this effort does not fully address the challenge of deploying DNNs effectively in memory-scarce platforms, which often necessitates specific scheduling/tiling and remains an open problem.

Going beyond existing approaches, this paper introduces and demonstrates a visual navigation engine capable of sophisticated workloads, such as real-time CNN-based autonomous visual navigation, entirely operating within the limited power envelope of nano-scale UAVs (approximately 0.2W). Previously, such autonomous navigation functionalities were confined to standard-sized UAVs, typically equipped with power-intensive processors (≥ 10W) or relying on external processing and sensing, such as GPS. The proposed system overcomes these limitations by utilizing an onboard ultralow-power processor and a learning-based navigation approach.

III. Methodology

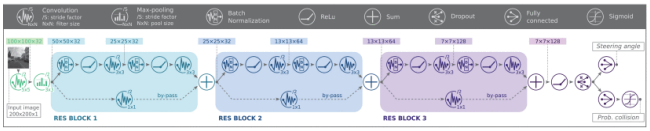

Fig. 1. Dronet

A. DroNet

DroNet is a compact residual Convolutional Neural Network (CNN) architecture designed for the safe autonomous navigation of a quadrotor in diverse indoor and outdoor environments. The network predicts both the steering angle and collision probability to ensure the drone's secure flight. Inspired by residual networks and optimized for minimal image processing time, DroNet's topology is depicted in Figure 1. The steering and collision prediction tasks share residual layers to reduce network complexity and frame processing time. Two distinct fully connected layers then independently predict steering and collision probabilities. Training employs Mean Squared Error (MSE) for steering and Binary Cross-Entropy (BCE) for collision probability. To handle differing gradient magnitudes, a temporally dependent weighting of the two losses ensures training convergence. Additionally, hard negative mining is employed in the later stages of learning to focus optimization on challenging samples. Notably, the steering angle prediction leverages the Udacity dataset, while the collision probability prediction is trained with the Zürich bicycle dataset.

The outputs generated by DroNet serve as directives for the UAV's movement within a plane, determining the forward velocity (vk) and steering angle (θk). Specifically, the low-pass filtered collision probability influences the UAV's forward

velocity, while the low-pass filtered steering angle is translated into the drone's yaw control. This results in a streamlined network that processes visual data and directly generates control commands for the flying drone. The end-to-end learning of the relationship between perception and control offers several advantages, including a simplified and lightweight system with high generalization capabilities. Notably, the method has proven effective not only in urban environments but also in various new application spaces, demonstrating adaptability without the need for prior knowledge.

A. Gap 8 Architecture

Our primary computational platform for executing the majority of the DroNet tasks is the GAP8, which is a commercial embedded RISC-V multicore processor derived from the PULP open-source project. The GAP8 consists of an advanced RISC-V MCU (Microcontroller Unit) combined with a programmable octa-core accelerator. This accelerator incorporates RISC-V cores specifically enhanced for digital signal processing and embedded deep inference.

The detailed architecture of GAP8 is illustrated in Figure 2. The processor comprises two distinct power and clock domains: the FABRIC CTRL (FC) and the CLUSTER (CL). The FC is an advanced MCU that includes a single RISC-V core paired with 512 KB of SRAM (L2 memory). This core utilizes an in-order, DSP-extended four-stage microarchitecture that implements the RISC-V instruction set architecture. The core supports the RV32IMC instruction set, encompassing standard ALU instructions along with multiplication instructions, and it can execute compressed code.
Furthermore, the core is expanded to incorporate a register–register multiply-accumulate instruction, packed single instruction multiple data (SIMD) DSP instructions (such as fixed-point dot product), bit manipulation instructions, and two hardware loops. Additionally, the SoC features an autonomous multichannel I/O DMA controller (µDMA) capable of transferring data between different components.

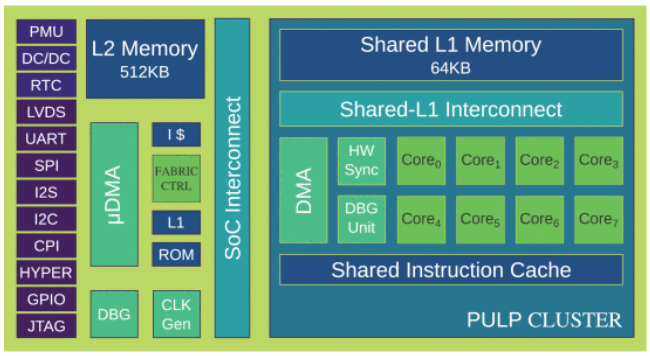

Fig. 2. Gap 8 Architecture

The GAP8 processor boasts a comprehensive array of peripherals, including QSPI, I2S, I2C, HyperBus, and a camera parallel interface. The L2 memory operates independently of the FC (FABRIC CTRL), and interfaces like HyperBus and QSPI allow connectivity with external DRAM or flash memory, effectively expanding the memory hierarchy with an external L3 offering a bandwidth of 333MB/s and a capacity of up to 128Mbit. Additionally, the GAP8 SoC incorporates a dc/dc converter on-chip to convert battery voltage to the required operating voltage. Two separate frequency-locked loops (FLLs) are also included for ultralow-power clock generation.

The CL (Cluster) is dedicated to accelerating computationally intensive tasks and comprises eight RISC-V cores (similar to the FC core). These cores share a 64 KB multibanked shared L1 scratchpad memory through a low-latency, high-throughput logarithmic interconnect. The shared L1 memory allows single-cycle concurrent access from different cores, minimizing contentions in memory-intensive operations. Instruction streams for the eight cores are sourced from a single shared, multiported I-cache to enhance energy efficiency in data-parallel code.

A cluster DMA controller facilitates data transfer between the shared L1 scratchpad and the L2 memory, supporting 1-D and 2-D bulk memory transfer on the L2 side (only 1-D on the L1 side). A dedicated hardware synchronizer enables fast event management and parallel thread dispatching/synchronization, enabling ultrafine grain parallelism on the cluster cores. The CL and FC share a single address space and communicate via two 64-bit AXI ports, one in each direction. A software runtime residing in the FC oversees tasks offloaded to the CL and the µDMA. During operation, a low-overhead runtime on the CL cores utilizes the hardware synchronizer to implement shared-memory parallelism in the style of OpenMP.

IV. DNN Mapping Methodology

A. Deep Neural Networks

Deep Neural Networks (DNNs) represent a sophisticated class of artificial neural networks designed to automatically extract hierarchical features from input data. Characterized by multiple hidden layers between the input and output layers, DNNs leverage these layers to learn increasingly abstract representations as data passes through the network. Each layer contains interconnected neurons that apply weighted transformations to the input, introducing non-linearities through activation functions. The depth of DNNs enables them to capture intricate patterns and dependencies in data, making them particularly effective in handling complex tasks.

Training a DNN involves adjusting the weights and biases of its connections using a process called backpropagation. During training, the network learns to minimize the difference between predicted outputs and actual targets by iteratively updating its parameters. This optimization is guided by a chosen loss function that measures the disparity between predictions and ground truth. The capacity of DNNs to learn from large datasets, paired with their ability to model non-linear relationships, has contributed to their remarkable success in various domains.

Deep learning, a subset of machine learning, heavily relies on DNNs to automatically discover intricate features and representations from raw data. Applications of DNNs span diverse fields, including image and speech recognition, natural language processing, computer vision, and more. Their versatility and ability to generalize patterns from data make them a go-to choice for solving complex problems.

Despite their effectiveness, training deep neural networks poses challenges. Computationally intensive tasks and the need for substantial labeled data are common hurdles. Overfitting, where the model becomes too specialized in the training data, is another concern, often mitigated through regularization techniques. Nonetheless, the capacity of DNNs to model complex relationships and learn from large datasets continues to drive their prominence in modern machine learning and artificial intelligence applications.

V. Experimental findings

Three key parameters guided our assessment of the visual navigation engine: achieving real-time deadlines, staying within power limits, and maintaining accurate closed-loop control in the face of unforeseen impediments. These evaluations were carried out using the PULP-Shield setup. Extensive testing was done on the GAP8 to investigate differences in core voltages (1.0 V, 1.2 V) and frequencies ranging from 50 to 250 MHz with regard to performance and power consumption. These studies emphasized configurations that performed exceptionally well in terms of energy efficiency, with a focus on the benefits that were seen when running at 1.0 V.

The power distribution throughout the system as well as the influence that the PULP-Shield brought about were carefully examined. Crucially, DroNet could be executed while respecting recommended power restrictions because the module's power consumption stayed well within the defined power envelope. An analysis of the operating lifetime showed that it had been shortened by about 22% from the initial lifetime. This was mostly because of the extra weight and computational burden that the PULP-Shield and DroNet execution imposed.

Our solution, which directly operated on Fixed16, had improved performance when compared to CMSIS-NN, a top microcontroller CNN implementation. This led to a real throughput of 2.81 MAC/cycle, which is a notable improvement over the highest performance of CMSIS-NN. Additional comparisons with high-performance microcontrollers, like the STM32H717, highlighted the comparable performance levels our system attained with significantly lower power consumption. This confirmed that our parallel-ultralow-power method worked as intended.

We also carefully examined our system's responsiveness to sudden impediments in order to assess control accuracy. Even at low frequencies, the combined system outperformed the original design [2] in its ability to recognize and respond to obstructions. By carefully optimizing the network and processing up to 18 frames per second, our system showed that it could take advantage of the platform's mobility to navigate both indoor and outdoor spaces and successfully avoid dynamic impediments.

VI. CONCLUSION

The Internet of Things (IoT) has enormous potential for using nano- and pico-sized UAVs as essential components. They are perfect for a variety of applications, including smart sensor and IoT hub mobility, surveillance, and inspections. Their small size and agility make them highly maneuverable. However, these UAVs need advanced capabilities to operate independently in complex contexts such as dangerous locations or metropolitan streets.

In this research, we present a complete deployment mechanism designed to allow the direct execution of resource-constrained mW-scale nodes with complicated deep learning algorithms. Our work is the first completely integrated visual navigation engine, combining hardware and software, designed specifically for autonomous nano-UAVs, allowing for full onboard computation. With this feature, these UAVs can function independently in situations where the additional power consumption or latency of wirelessly connected centralized systems are too costly.

Our system allows real-time computation of DroNet, an advanced CNN-based autonomous navigation algorithm. It is based on the Crazyflie 2.0 nano-UAV and the GreenWaves Technologies GAP8 SoC combined with the STM32 MCU. Using the most energy-efficient SoC configuration, experimental results show a performance of 6 frames/s at 64 mW, which scalable up to 18 frames/s within an average power budget of 284 mW for computing. Significantly, in comparison to the baseline system—a commercially available standard-size UAV connected to a remote PC running DroNet at 20 frames per second—this performance is attained without sacrificing the quality of results.

Our results validate that both systems are capable of quickly identifying impediments, which guarantees safe flight even at very high speeds, such the Crazyflie 2.0's 4 m/s. In order to enhance the potential of autonomous nano-UAVs as mobile smart sensors connected to the Internet of Things, we give back to the community by making our PULP-Shield design, together with all related code, datasets, and trained networks, available as open-source resources.

This work provides insights and tools that open up a wide range of advanced applications in the field of IoT-connected mobile smart sensors, which is a key step toward realizing the potential of autonomous nano-UAVs.

References

[1] N. H. Motlagh, T. Taleb and O. Arouk, "Low-altitude unmanned aerial vehicles-based Internet of Things services: Comprehensive survey and

future perspectives", *IEEE Internet Things J.*, vol. 3, no. 6, pp. 899-922, Dec. 2016.

[2] A. Loquercio, A. I. Maqueda, C. R. del Blanco and D. Scaramuzza, "DroNet: Learning to fly by driving", *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1088-1095, Apr. 2018.

[3] F. Conti et al., "An IoT endpoint system-on-chip for secure and energy-efficient near-sensor analytics", *IEEE Trans. Circuits Syst. I Reg. Papers*, vol. 64, no. 9, pp. 2481-2494, Sep. 2017.

[4] D. Palossi, A. Gomez, S. Draskovic, A. Marongiu, L. Thiele and L. Benini, "Extending the lifetime of nano-blimps via dynamic motor control", *J. Signal Process. Syst.*, vol. 91, no. 3, pp. 339-361, Feb. 2018.

[5] D. Floreano and R. J. Wood, "Science technology and the future of small autonomous drones", *Nature*, vol. 521, no. 7553, pp. 460-466, May 2015.

[6] Z. Liu, Y. Chen, B. Liu, C. Cao and X. Fu, "HAWK: An unmanned mini-helicopter-based aerial wireless kit for localization", *IEEE Trans. Mobile Comput.*, vol. 13, no. 2, pp. 287-298, Feb. 2014.

[7] M. Piccoli and M. Yim, "Piccolissimo: The smallest micro aerial vehicle", *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 3328-3333, May 2017.

[8] Y. Lin et al., "Autonomous aerial navigation using monocular visual-inertial fusion", *J. Field Robot.*, vol. 35, no. 1, pp. 23-51, Jul. 2017.

[9] D. Falanga, E. Mueggler, M. Faessler and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision", *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 5774-5781, May 2017.

[10] G. Loianno, D. Scaramuzza and V. Kumar, "Special issue on high-speed vision-based autonomous navigation of UAVs", *J. Field Robot.*, vol. 35, no. 1, pp. 3-4, 2018.

[11] Y. Yang, Z. Zheng, K. Bian, L. Song and Z. Han, "Real-time profiling of fine-grained air quality index distribution using UAV sensing", *IEEE Internet Things J.*, vol. 5, no. 1, pp. 186-198, Feb. 2018.