OPEN ACCESS
Journal of Astronomical Instrumentation, Vol. 12, No. 1 (2023) 2340008 (21 pages)

DOI: 10.1142/S2251171723400081

World Scientific
www.worldscientific.com

Integrated Computing Platform for Detection and Tracking of Unidentified Aerial Phenomena (UAP)

Richard Cloete[1,2,*], Phillip Bridgham[2], Sergei Dobroshinsky[2], Carson Ezell[2], Andriy Fedorenko[2], Frank Laukien[2,3], Sarah Little[2,4], Abraham Loeb[1,2], Eric Masson[2], Matthew Szenher[2], Wesley Andrés Watters[2,5] and Abigail White[1,2]

[1]Department of Astronomy, Harvard University, 60 Garden Street, Cambridge, MA, USA 02138

[2]Galileo Project, 60 Garden Street, Cambridge, MA, 02138, USA

[3]Department of Chemistry and Chemical Biology, Harvard University, 12 Oxford Street, Cambridge, MA, USA 02138

[4]Scientific Coalition for UAP Studies, Fort Myers, Florida, USA

[5]Deptartment of Astronomy, Wellesley College, 106 Central St., Wellesley, MA 02481, USA

*richardcloete@cfa.harvard.edu

Received August 6, 2022; Revised May 16, 2023; Accepted May 23, 2023; Published August 19, 2023

The Galileo Project aims to shed light on the nature and characteristics of Unidentified Aerial Phenomena (UAP). We are developing a multi-modal instrumentation suite that will monitor the sky in seven electromagnetic and three audio bands. Computing will play a critical role in this project, enabling the automated collection and processing of data. In this paper, we provide a brief overview of data sources, and describe our plan for computing infrastructure and architecture. We present a proposed real-time pipeline for distinguishing between natural and human-made phenomena, and for detecting objects that fall outside the phenomenological envelope of known phenomena. In addition, we outline the algorithms we will test and evaluate for use in offline data analysis. While preliminary, our work represents a significant step towards a unified data capture and analysis platform for the systematic detection and rigorous scientific study of unusual aerial phenomena in a regional airspace.

Keywords: Aerial anomaly; anomaly detection; tracking; UAP; UFO; unidentified aerial phenomena; unidentified aerospace phenomena; unidentified anomalous phenomena; object detection; object classification; deep learning; computer vision; system architecture; edge computing; data processing; synthetic data.

1. Introduction

In recent years, there has been a marked increase in attention given to the topic of Unidentified Aerial Phenomena (UAP) — often referred to as Unidentified Flying Objects (UFOs). In 2021, the U.S. Department of National Intelligence recognized the

*Corresponding author.

presence of objects with unusual flight capabilities operating within restricted airspace, and has called for better data collection to help understand UAP, their prevalence, nature, and the risks they pose to national security (ODNI, 2021). Shortly after, an amendment to the National Defense Authorization Act (NDAA) for the Fiscal Year 2022 mandated that the Director of National Intelligence submit (within 180 days of the enactment of the NDAA) an unclassified report on UAPs (Gillibrand, 2021; Rubio *et al.*, 2021). More recently, the U.S. Congress voted to make it easier to report UAP incidents by establishing a secure reporting system, and promised to protect former officials from reprisal should they wish to come forward and share classified UAP-related information with congressional armed services and intelligence committees (Bender, 2022). Further motivations for the study of UAP, the goals of the investigation, and traditional arguments typically raised in opposition to such work, are addressed in Watters (2023).

The Galileo Project (GP) was established in the summer of 2021 to investigate anomalous aerospace phenomena including ambiguous interstellar objects (ISOs) such as Oumuamua (Siraj *et al.*, 2022) and UAP (Loeb, 2022). The project team is currently designing a multi-sensor instrument suite to perform a broadly scoped census of aerial phenomena in the U.S. Our initial instrument suite includes one passive radar array, one eight-camera infrared (IR) array, one optical pan–tilt–zoom camera, two all-sky cameras (one optical and one optical/NIR), one three-band (infrasonic, ultrasonic, audible) acoustic array, a broadband radio frequency (RF) detection system, as well as a terrestrial and space weather sensor package with ultraviolet (UV), magnetic field, and particle detection (see Sec. 2). The instruments are being tightly calibrated and synchronized so as to ensure minimal interference from background/environmental signals, and to ensure that observations are made across multiple independent sensors along a unified timeline. Together, our instruments will continuously monitor the sky and immediate environment across much of the electromagnetic spectrum. A processing pipeline that leverages artificial intelligence (AI) will be used to analyze these data streams, enabling real-time detection, tracking, classification and filtering of airborne objects. In the process, these will distinguish known objects from statistical outliers. In this way, we aim to separate — in real time — known biological, atmospheric, and technological phenomena (e.g. birds, sprites, aircraft, etc.) from those that appear either to operate outside of the standard performance envelope of known objects or phenomena, or which exhibit novel characteristics or properties. Note that the terms 'object' and 'phenomena' will be used interchangeably, unless otherwise specified, as this will allow us to discuss 'things' in the sky without without ascribing particular characteristics or properties. We will use 'outlier' to describe observations or measurements that deviate significantly from the mean. Importantly, a statistical outlier is distinct from the idea of a 'scientific anomaly' as defined in Watters (2023), which refers specifically to a corroborated, statistically distinctive phenomenon that resists explanation in terms of prevailing scientific beliefs. For more information on our instruments, data sources, and study methodology, see our investigation overview paper: Watters (2023).

Computing will play a central role in the processing, aggregating, analysis (both in real-time and in hindsight) and storing of data. The present paper is meant to provide insight into our initial Phase 1 approach to computing; the system will evolve as issues are encountered and as lessons are learned. (See Watters (2023) for a description of project phases.) The first part of this paper (Sec. 2) provides an overview of our instrumentation, providing context to the sources and types of data that will be processed. In Sec. 3, we discuss our computing infrastructure, including hardware employed, networking, storage and communications. Section 4 describes how we intend to detect and track potential UAP. Section 5 presents some of the ways in which we intend to analyze data off-line — after potential UAP events have been recorded. Ensuring that our data is reliable and tamper-free will be important for drawing robust conclusions and as such, Sec. 6 briefly describes our data security approach.

2. Instrumentation and Data Sources

The Galileo Project is building a multimodal and multispectral suite of calibrated scientific instruments to monitor the sky across wide bands of the electromagnetic and acoustic spectrum. In addition to low-cost portable systems, the project is developing a stationary 'observatory-class' system that will monitor the airspace from one primary site and several secondary sites. The Phase 1 observatory-class

system will consist of six instruments that will be used to make corroborative detections and measurements of objects in the regional airspace. A detailed summary of the instruments can be found in Watters (2023).

This instrumentation requires a computing infrastructure to capture data, perform real-time analysis and outlier detection, drive narrow-field sensors for observation of targets of interest, provide communications between the various system components, and to egress data to the cloud for hindsight review. This section briefly describes our instrumentation to provide context regarding the sources of data our system will address, and then describes the computing hardware that will facilitate system operation.

2.1. *Wide-field cameras*

Wide-field cameras will be used for targeting and tracking of aerial objects across multiple wavelengths, forming the basis of our 'detection' strategy. We aim to cover as much of the sky as possible, imaging in infrared, near-infrared and visible electromagnetic bands. Presently, we have two wide-field camera systems: the *Dalek*, and the *Alcor*.

The Dalek is a fiberglass dome housing seven 50° Long Wave Infrared (LWIR) FLIR Boson 640 cameras arranged in a hemispherical configuration with one 95° LWIR camera at the zenith, providing all-sky coverage in thermal infrared. Each camera operates at 30–60 frames per second (FPS) and captures 16-bit images at a resolution of 640 × 512 pixels. In addition to the LWIR cameras, a wide-angle (150°), ZWO ASI462 camera provides all-sky coverage in the visible/Near Infrared (NIR) spectrum. This NIR camera operates at 136 FPS, each with a resolution of 1936 × 1096.

The Alcor OMEA 9C is an all-sky optical camera that captures imagery at 3 FPS, each with a resolution of 9575 × 6380 pixels. This camera, like the LWIR and NIR cameras mentioned, will be used for object detection. In addition, all-sky camera detections will be used to define the duration of 'events' and will be used for targeting narrow-field instruments. When additional all-sky cameras are deployed at secondary sites (separated by several km from the primary site (Szenher, 2023)), the angular coordinates of these detections can be used to determine triangulated position.

2.2. *Narrow-field cameras*

These cameras will be directed towards targets of interest in order to analyze their features, spectra, polarimetry, and photometry. Our instrumentation suite currently employs a single Beacon 8 narrow-field, Pan–Tilt–Zoom (PTZ) IP camera with Wide Dynamic Range (WDR) and 40× optical zoom. It has a frame rate of 30 FPS, a 3840 × 2160 pixel resolution, and a bit-depth of 16. The Beacon 8 will be the only narrow-field camera in our Phase 1 observatory-class system, but we intend to add other narrow-field cameras based in future project phases in order to achieve 1 arcsecond/pixel resolution.

2.3. *Passive Radar*

We are developing an omni-directional or (optionally) beam-formed multi-static passive radar system consisting of an array of antennas and receivers, called 'Skywatch'. Passive radar has a significantly larger detection range (≈ 150 km) than any of our wide-field cameras, and is not impacted by cloud cover. It works by detecting the reflection of ambient radio signals, such as television or radio broadcasts, from moving targets, such as aircraft or ships. The reflected signal is then processed to extract information about the target, including its position, velocity, and size, providing more kinematic information about aerial objects than is possible with camera arrays. The system under development has a 3-channel receiver, which will record 300 s of narrow-band FM radio station data at a sample rate of 1 MHz; this procedure will be repeated every 1024 s (17 min), or approximately 85 times per day. In Phase 1 of the project, Skywatch will only be used to measure the range and motion of objects detected by our wide-field cameras. In a future iteration of our system, we intend to use Skywatch for both detection and tracking of aerial objects. More information about the design of this instrument can be found in Randall (2023).

2.4. *Acoustics*

The Acoustic Monitoring Omnidirectional System (AMOS) is a passive, omnidirectional, multi-band suite of microphones that capture ambient sounds in the infrasonic, ultrasonic and audible spectrum. Data sampling rates for the infrasonic, audible and ultrasonic are 50 Hz, 44.1 kHz, and 512 kHz, respectively.

All three instruments will run continuously. In Phase 1, data captured by AMOS will be used primarily to correlate and verify detections made by our wide-field and narrow-field instruments. In a later phase of the project, we intend to explore object detection, classification, and triangulation using acoustic signals. More information on AMOS can be found in Mead (2023).

2.5. *Radio spectrum analyzer*

We will also measure and characterize radio and microwave emissions via a radio spectrum analyzer attached to a wide-band antenna; this system is called 'Spectre'. A spectrum analyzer works by measuring the power of RF signals across a range of frequencies. Measurements can reveal the presence of signals that indicate the presence of specific types of objects, such as the radar signals emitted by aircraft, or the telemetry signals transmitted by satellites. By analyzing the frequency, power, and modulation of these signals, we may be able to glean valuable information about the objects detected in the sky (Kay & Marple, 1981). In addition to this, Spectre will help us detect and locate sources of RF interference, thus improving the robustness of our instrumentation suite and the veracity of our data.

2.6. *Environmental sensors*

The New PArticle Counter k-Index Magnetic ANomaly (NPACKMAN) platform was developed specifically for the Galileo Project. This instrument will monitor local meteorological conditions such as wind velocity, temperature, humidity, pressure, and cloud cover, and will also measure magnetic fields, energetic particle counts and ultraviolet radiation. The data will be used to perform offline analysis to identify signatures or patterns corresponding to events of interest within the environmental data. The analysis will help with further characterization of the detected objects as well as for the calibration of signals, such as sound or light attenuation. A Raspberry Pi 4 drives the NPACKMAN's sensors and manages data collection.

2.7. *External data*

In addition the above, we will utilize a range of external data sources for monitoring the regional airspace for known signals. In particular, we intend to collect Automatic Dependent Surveillance–Broadcast (ADS-B) data. ADS-B data includes the position, altitude, speed and other parameters pertaining to aircraft flight characteristics (airplanes, helicopters, gliders, etc.) within a certain radius (Schfer *et al.*, 2014). In a later phase of the project, we will explore the use of other external data such as information concerning regional weather, geomagnetic field variations, and seismicity, to assist with the interpretation of the data we directly collect.

Together, these instruments will produce a wide range of data types at significant volume. In Secs. 4 and 5, we discuss plans for processing and analyzing data from these instruments, but first we describe the computing infrastructure that will drive our instrumentation.

3. System Architecture

The instrumentation described in Sec. 2 requires a system architecture that supports the capture and processing of data, executes real-time analysis, drives narrow-field sensors towards observation of targets of interest, mediates communications between system components, and transfers raw and reduced data products to the cloud for hindsight review. This section describes the system architecture that will support Phase 1 of the project and how sub-systems will be integrated to achieve the goals of the Phase 1 observatory. At the foundational level, the GP system architecture that is currently in development can be described as a high-performance, high-throughput data processing system inspired by a hybrid event-driven and distributed-system architecture. We aim to build a system that is decentralized by nature, and, due to real-time requirements, favors localized computing over centralization — both in terms of data and computation. This approach will ensure the shortest and fastest feedback loop between data-processing logic and sensor control, thereby improving system adaptability and reaction-time — features important for responding in real time to potentially anomalous events.

From a functional point of view, all of the GP system activities — including security, communication, automation, computation, data management, and monitoring — can be categorized into three core behavior groups: System Support Services, Edge Computing, and Cloud Computing. Together, these groups will ensure proper separation of concern while defining clear responsibility and functional boundaries (Beck *et al.*, 2001). Figure 1 illustrates our proposed system architecture. Security is discussed in Sec. 6.

3.1. *System support services*

System Support Services will provide common services required within and across all systems and functions. These services will include time synchronization, task scheduling for automation, remote management, network and telecommunications, data synchronization and management, hardware and systems monitoring, and security (see Sec. 6 for more information) and auditing services to maintain secured and predictable communications.

3.1.1. *Time synchronization*

To ensure reliable and consistent data management within the GP systems, maintaining time synchronization across all computer systems and devices will be critical. Precise time synchronization is essential for the proper functioning of critical real-time applications, control and measurement systems, and video networks. It is also necessary to establish frequency, phase, and time synchronization between distributed nodes and to correlate measurement and event data distributed across systems. Fundamentally, precise time synchronization is a requirement for optical, IR, and acoustic source tracking, triangulation, and object localization. To achieve this, we will implement a comprehensive time synchronization strategy that relies on the network time protocol (NTP). This protocol is widely used and requires only internet access or a local NTP server that is synchronized with a global NTP service. In addition to the internet-based NTP service, the GP system will also have access to Global Positioning System (GPS) time synchronization through its telecommunication equipment. The proposed NTP time synchronization strategy will provide millisecond precision, with provisions for future expansion to PTP/IEEE 1588 (Precision Time Protocol) with nanosecond precision. The importance of time synchronization in a distributed system like the GP systems cannot be overemphasized. It ensures consistency and accuracy of data across all nodes, thereby improving system reliability and performance.

3.1.2. *Network and telecommunications*

The Phase 1 system is projected to generate approximately 120–250 GB per day. A significant portion of this data will need to be transmitted to the Cloud and other remote storage systems for testing and validation purposes, particularly in the initial months following deployment. In addition, remote access to instrumentation and computing infrastructure will be needed to perform system status checks, service restarts, and software updates.

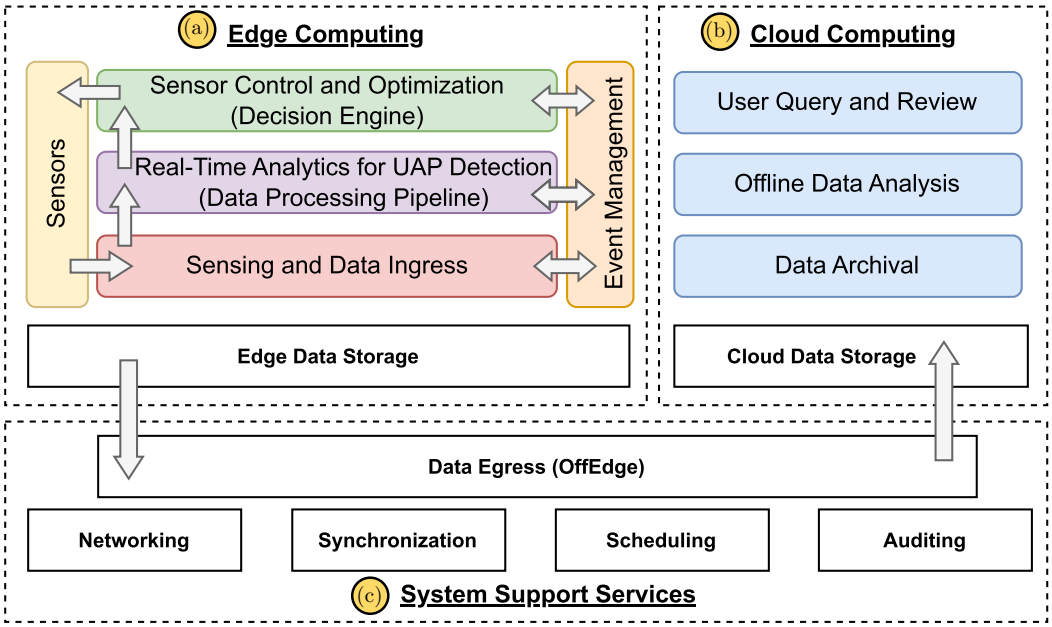

Fig. 1. (Color online) Proposed computing system architecture illustrating three sub-systems: Edge Computing, where real-time data acquisition and processing takes place; Cloud Computing, where data archival and offline analysis is performed; and System Support Services, which coordinates system activities such as data synchronization, scheduling of tasks, performs data egress, manages networking, and ensures data tractability via auditing.

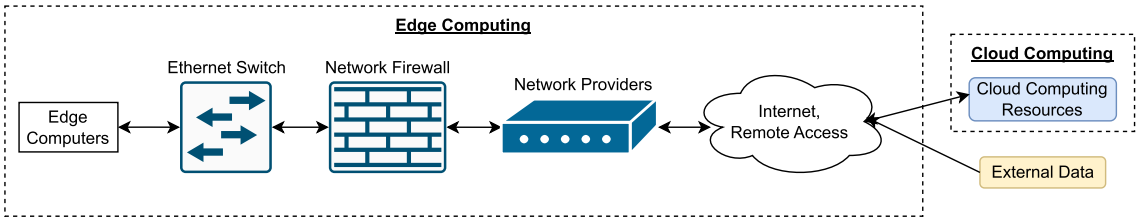

Fig. 2. (Color online) Overview of planned GP Networking.

Combining data egress and remote access activities on a single internet connection poses a challenge, as there is a risk that the bandwidth requirements of data egress may impact remote access activities. To overcome this challenge and ensure a constant, reliable, and cost-effective internet connection, the GP system network will rely on equipment enabled with Service-Defined Wide-Area-Network (SD-WAN) technology (Yang *et al.*, 2019). Figure 2 shows a high-level overview of our planned network design.

SD-WAN technology provides network scalability and enables network-traffic load-balancing across multiple internet service providers with different network topologies, including cellular/5G and satellite/StarLink. While commercially available 5G plans offer fast, dependable internet connectivity in and around suburban areas, their monthly data upload allowance is limited, and their upload rate is restricted. Therefore, 5G will primarily be used for remote system checks, updates, and other similar tasks. Starlink, although more expensive than 5G plans, offers internet access in remote areas, including at sea, and without data limitations and as such will serve as the primary means for transmitting data to the Cloud and other remote storage systems. Our current Starlink package provides upload rates of 8–25 Mbps. Considering a conservative 15 Mbps, we will be able to egress approximately 158.16 Gigabytes/day. To ensure sufficient bandwidth, we will likely need to employ multiple Starlink nodes, though for this initial phase, a single Starlink may be sufficient.

In the subsequent stages of this project, it is anticipated that the daily data generation will approximate 9–10 Terabytes (TB). Out of this aggregate data volume, nearly 8.5 TB will be constituted by the input from both wide-field and narrow-field cameras, as illustrated in Table 1. The sheer magnitude of such data presents a formidable challenge.

Given the current network conditions and the assumption of continuous operation, it is estimated that approximately 58–65 Starlink connections would be necessary to manage this volume of data. However, it is essential to note that achieving optimal network conditions consistently is improbable, and that the cost (and required deployment space) of operating so many nodes would be prohibitively expensive. Therefore, it will be important for us to explore alternative strategies. One potential strategy is the selective egress of data centered around events of interest. This strategy would involve capturing data from e.g. 2 min preceding an event, the entire duration of the event itself, and 2 min following the event. By adopting this approach, we can effectively filter out 'routine' or 'non-

Table 1. Data production.

Instrument/source	TB/day
Dalek (7 Flir Bosons, 1 × ZWO ASI462, 1 × Alcore OMEA 9c)	8.79
Beacon 8.0 PTZ camera	0.19
Skywatch (1 tower)	0.15
AMOS	0.10
NPACKMAN	0.000049
Spectre (1 tower)	0.29
Total data	**9.52**

critical' data, thereby managing the data load more efficiently. Additionally, the implementation of data compression techniques will be crucial. However, the degree of compression applied will be contingent on the nature of the data and the requisite level necessary to ensure that the subsequent analysis remains scientifically valid and robust.

3.1.3. *Data synchronization and management*

It is imperative to establish robust data synchronization mechanisms that ensure the integrity and traceability of every file from its creation to its final destination in cloud storage, and throughout its transfer. To accomplish this, a computer program known as OffEdge is currently under development. This program will be deployed on all computers involved in either the production or transfer of data. OffEdge will perform several critical functions, including tracking every file's creation, generating checksums to verify data integrity, and managing file synchronization between the Edge and remote servers such as cloud or remote network-attached storage. This will be accomplished through the implementation of a consistent and timely data persistence and backup policy, which will be enforced by a file-watcher trigger for rules-based processing.

3.2. *Edge computing*

The Edge Computing sub-system will consist of data storage, sensors, data processing pipeline, sensor control and optimization, and event management components (see Fig. 1(a)). In its entirety, the Edge Computing sub-system will provide all the real-time data acquisition and processing functionality required for a functioning observatory. The Edge Computing hardware that will be employed during Phase 1 currently consists of several 'edge computers'. Edge computers are small, decentralized computing devices that are located at or near the edge of a network, where they can collect and process data from the physical world. This edge-computing approach will be adopted so that data processing and analytics can be performed locally at the site of the instrumentation, reducing the need for data to be sent to a central data center or the cloud. Thus, the approach will improve the speed and efficiency of data processing, reduce latency, and enable real-time decision-making. The Phase 1 edge computer 'inventory' consists of several NVIDIA Jetsons (each equipped with a minimum of 6 CPU cores, 8–16 GB of RAM, a 384-core GPU and 1 TB SSD storage), Raspberry Pi 4 units, and Intel NUC's. Each of these edge computers is connected to at least one instrument, providing the processing power required to perform tasks such as data capture, real-time analysis (including computationally intensive AI tasks see Sec. 4), controlling actuators (e.g. in order to shield sensitive equipment from direct sunlight), communications between devices, data transfer to local and remote storage systems, etc. The Sensors (Fig. 1) represent the sensors connected to the observatory system, which are described in Sec. 2. The remainder of this section describes the Sensing and Data Ingress functionality that receives raw sensor data and prepares the data for subsequent processing.

The edge computers are designed to perform AI-related tasks at-the-edge (Mittal, 2019), making them ideal for running object detection and classification models. Indeed, we will primarily use NVIDIA Jetsons to drive the capture of video data from various LWIR, NIR, and visual cameras (both wide-field and near-field), and to perform local computer vision tasks (see Sec. 4). We will also employ an NVIDIA Jetson for the processing of data acquired by our Skywatch passive radar instrumentation. This is because, like computer vision tasks, processing passive radar signals is computationally intensive (more-so than those of traditional active radar signals, as it requires advanced signal processing techniques to isolate the target signal from the background noise) (Szumski *et al.*, 2009; Randall, 2023). Table 2 summarizes the specification of our edge computers. To house the edge computers, we have modified a 12U desk-side server rack-mount. This 'edge enclosure', which can be see in Fig. 3(a), is designed to protect our computing hardware from the elements, while two fans provide ventilation. The figure shows four USB hubs (2 on the front; top shelf), three of our NVIDIA Jetsons (middle), and power supply (including a dedicated UPS) equipment (bottom). In addition to the edge computers, we will also employ a dedicated ThinkMate server which will be housed off-site. This server will be used to perform AI model development, and offline data review and analysis (see Sec. 5). Figure 3(b) depicts our dedicated ThinkMate server, while its specifications are illustrated in Table 2.

The Real-Time Analytics for UAP Detection component (Fig. 1(a)), also referred to as the Data Processing Pipeline, encapsulates the core GP

Table 2. Specifications for edge computers and our thinkmate server.

Metric	Nvidia Jetson	Raspberry Pi 4	Intel NUC	ThinkMate Server
CPU	6-core NVIDIA Carmel ARM® v8.2 64-bit CPU, 6MB L2 + 4MB L3	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz	Intel Core i5-1135G7 2.40 GHz Processor (11th Gen, upto 4.2 GHz, 8 MB Cache, 4-Cores)	Sixteen-Core AMD Ryzen Threadripper PRO 3955WX Processor - 3.90 GHz 64MR L3 Cache (280W)
RAM	8 GB 128-bit LPDDR4x, 59.7 GB/s	8 GB LPDDR4-3200 SDRAM	32 GB DDR4 SO-DIMM	128GB PC4-25600 3200 MHz DDR4 ECC RDIMM
Graphics	384-core NVIDIA Volta™ GPU with 48 Tensor Cores	N/A	Integrated Graphics Integrated Graphics	NVIDIA RTX A6000 Graphics card - 48GB GDDR6 - PCIe 4.0
Storage	16 GB eMMC 5.1, with 1TB external SSD	Micro-SD card	2TB PCIe SSD (Solid State Drive)	1.0TB Samsung 980 PRO M.2 PCIe 4.0 and 40TB SATA 6.0 Gb/s 7200RPM - 3.5″ - Ultrastar DC HC330 (512e)

capability of real-time object detection and classification. To support real-time analytics, this component design will be stream-based and will rely on high-performance edge computing devices such as NVIDIA Jetsons described in the prior section. The Phase 1 Data Processing Pipeline is discussed in length in Sec. 4. The output of this component will produce object detection events with classification data. This event information will be published to the Sensor Control and Optimization component via the Event Management component (Fig. 1(b)).

Due to the real-time nature of the Edge Computing sub-system processing and the goal of object detection (required for targeting narrow-field instruments), the Edge Computing sub-system will be based on an event-driven architecture that will require messaging and event management. Once the Data Processing Pipeline identifies an object of interest and has computed and correlated position and classification information, there potentially exists a short window of time to direct the narrow-field instruments (e.g. the Beacon 8 PTZ camera) to focus on to that target. Likewise, as the object moves, the PTZ component will require event updates to indicate the new position of the object of interest. The ability to provide this near-real-time event sharing will be provided by the Event Management component.

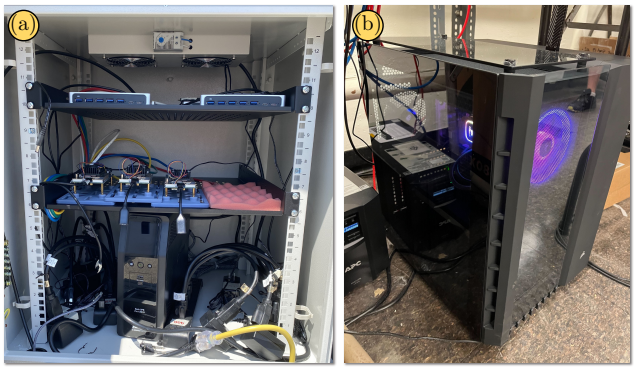

Fig. 3. (Color online) A; Edge computer enclosure with 3 × NVIDIA Jetson Xavier's, 2 × USB Hubs and an Uninterrupted Power Supply (UPS); these will be used for real-time aerial outlier detection at-the-edge. B; The ThinkMate server for offline data analysis.

To facilitate communications between edge computers, we will employ ZeroMQ–a mature, widely-used, open source, high performance, asynchronous messaging library (Rakhimov *et al.*, 2020; Kang *et al.*, 2020; Happ *et al.*, 2017). ZeroMQ is brokerless (i.e. middleware designed to 'manage' messages passed between endpoints), and as such, it is lightweight with little computational overhead which makes ZeroMQ fit-for-purpose for our edge computing goals. To meet scalability requirements, we will employ a publisher-subscriber pattern (zmq, 2021). The processes that generate data or events will 'publish' messages on predefined topics. The various processes that require receipt of data or an event will be configured to 'subscribe' to the predefined topics of interest at design time, to establish the GP event communications model. Each component process will utilize a shared library, which will provide the underlying ability to send or receive messages establishing GP event communications

between processes and across hosts. Given that the edge devices, operating systems, and code languages may vary between instruments, it will be important to ensure that messages share a standard, platform-agnostic encoding and message structure so that they can communicate effectively across heterogeneous technologies. To this end, we will use JavaScript Object Notation (JSON) (Severance, 2012) as this data exchange format. JSON provides a standardized technology agnostic, platform-neutral, and an extensible means for serializing and deserializing structured data.

The Sensor Control and Optimization component (a.k.a. Decision Engine or DE) will determine which of the objects detected by the Dalek and Alcor instruments should be targeted and tracked by narrow-field instruments such as the Beacon 8.0 PTZ camera. This component is discussed at length in Sec. 4.6.

3.3. *Cloud computing*

The Cloud Computing sub-system represents the cloud-based processing environment that provides the opportunity for batch processing without the constraints associated with remote on-premise edge environments. Due to the constraints with communication between the Edge Computing and Cloud Computing sub-systems, such as cost for utilization of the 5G service, only data for an event-of-interest will be transferred to the cloud. An 'event-of-interest' is a time sequence that includes one or more contiguous object detections. A buffer of time before and after an event-of-interest (e.g. 1 min before and 1 min after) will be applied to ensure a comprehensive analysis surrounding an event-of-interest data set. The egress of data into the Cloud Computing sub-system offers the benefit of applying various computation techniques, including the evaluation of alternative and new models, the enhancement of existing models, and general system calibration. To optimize the use of the cloud computing strategy we will investigate cloud-hosted data analysis tools and processing pipeline technologies, as well as techniques for online data visualization and end-user artifact review. Note that in Phase 1 of this investigation, *all* data products will be stored off-site (i.e. not in the Cloud) in order to ensure that we complete the comprehensive aerial census described in Watters (2023).

4. Phase 1 Data Processing Pipeline Overview

In order to detect and characterize UAP, the observatory system will be required to (i) operate autonomously and (ii) collect and process data in real-time. Autonomous operation is required because UAP events are, at present, unpredictable — they can happen at any time, without warning and at any location. As such, it is not practical to have a system that depends on a human operator. Data must be collected and processed in real time so that potential UAP can be detected, tracked and targeted by the narrow field instrumentation (e.g. the Beacon 8 PTZ camera). This section describes how we intend to perform real-time detection and tracking of potential UAP.

4.1. *Pipeline overview*

Figure 4 presents a high-level view of the Phase 1 real-time image data processing pipeline. The yellow boxes represent the data sources and instrumentation used (see Sec. 2), while the blue boxes depict edge computers used to process the incoming data. The white boxes with solid borders denote a processing operation, while the white boxes with dashed borders represent the communication of data.

As illustrated in the figure, images captured by the Dalek and Alcor instruments are recorded to a local edge computer. The data will be processed using computer vision (CV) models to perform object detection and classification (see Sec. 4.3). The

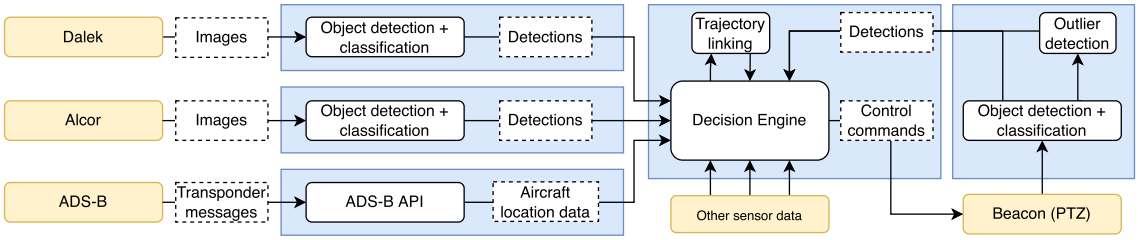

Fig. 4. (Color online) High-level view of the detection strategy.

CV module will output detection messages that contain information on the time of the detection, the position of the detected object (represented by x and y coordinates and the image *width* and *height*), the assigned classification, and the confidence level (expressed as a percentage) of the assigned class. Each detection message will be sent to the Decision Engine (DE), which will be hosted on a separate edge computer.

The DE will work by querying ADS-B data and analyzing the received detections to determine which of the detected objects require further investigation. That is, ADS-B data will be used to filter out detections that are likely aircraft, while classifications and their confidence scores will be used to filter out known objects. Detections of the same object captured across multiple frames will be combined; this will reduce the number of candidate UAP targets, while also providing 2D trajectory information (see Sec. 4.4). If the DE fails to determine, with high confidence, that an object being tracked is of known origin, or if he DE computes a high outlier score, then control commands will be sent to the Beacon PTZ, instructing it to orient itself towards and track the object of interest. The Beacon PTZ captures images of the object in color and at a much higher resolution than the Dalek and Alcor instruments. In addition, the Beacon PTZ has zoom and autofocus capabilities, which will allow us to captured images centered and focused on objects of interest (see Fig. 5 for a simulated example).

Images captured by the Beacon PTZ will be sent to two separate CV models: one to perform object detection and image classification, and another to determine whether the object being observed is novel (i.e. has not previously been observed by the camera) or extremely rare. The resulting detections will then sent be back to the DE, which will re-evaluate the object being tracked to determine if it remains of interest. Data pertaining to candidate UAP will be stored locally and then will be transferred to the cloud for further, offline analysis, which we discuss in Sec. 5.

4.2. *Synthetic data*

The first step towards being able to detect and classify objects in images is to train a suitable model. However, training models such as You Only Look Once (YOLO) (Jiang *et al.*, 2022) typically requires a significant number of labelled images. Crucially, the training data must be representative of the problem domain. This means that our training data should contain many examples of different objects (birds, aircraft, drones, etc.) of different shapes, sizes, colors, and at different angles, and under varying light and atmospheric conditions. No single such dataset exists and the few that may be

Fig. 5. (Color online) Sample narrow-field images rendered using AeroSynth.

relevant are often insufficient (i.e. lack enough data or fail to properly represent the problem at hand, are imbalanced, etc.). Further, datasets developed by third-parties often lack information regarding their collection process, accuracy, and veracity (Hittmeir *et al.*, 2019). Although we could manually collect and label our own data, this would be challenging as it would require that (i) we collect enough of it, (ii) samples are representative of the problem domain, and (iii) that classes ('categories' represented by a label) are balanced (i.e. they have the same number of samples per class so as to minimize classification bias).

Synthetic data has emerged as a promising and effective solution for the scarcity of appropriate machine-learning data, with numerous examples of its application in the literature (see Luo *et al.* (2019), Dewi *et al.* (2021) and Boikov *et al.* (2021)). In light of the labor-intensive, costly, and error-prone task of manual data collection, we developed *AeroSynth*, a synthetic image data generation tool built using Python and Blender (free and open-source 3D computer graphics software).

AeroSynth functions by importing 3D models and rendering them at arbitrary positions and orientations within the virtual 'sky' under realistic atmospheric conditions, lighting, and distortion. For realism, AeroSynth offers an option to include clouds and contrails in the rendered images. Images can be rendered in color or in monochrome infrared. Additionally, AeroSynth enables the configuration of virtual cameras to correspond with the specific parameters of physical cameras, allowing us to produce synthetic images that match the cameras used by Dalek, Alcor, or Beacon PTZ, ensuring that the data utilized to train our models is consistent with the conditions in which the models will operate in the real world. Every synthetic image produced is paired with a corresponding file containing the object class and bounding box coordinates for each object in the image.

AeroSynth offers two distinct modes, each with a unique purpose: 'capture sky' and 'capture target.' In 'capture target' mode, AeroSynth creates narrow-field images (see Fig. 5) of the sky, where the virtual camera is focused on a single target. This

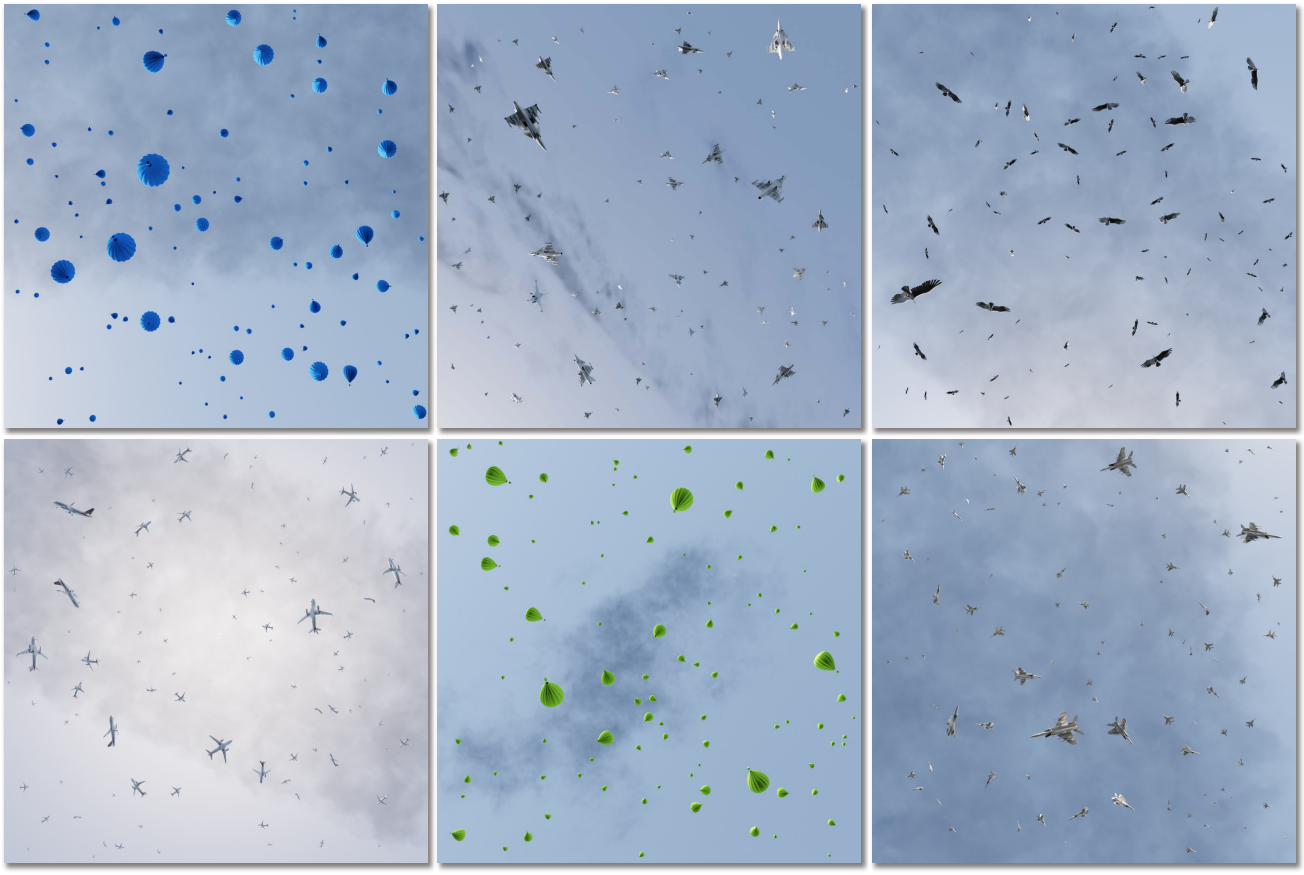

Fig. 6. (Color online) Sample wide-field images rendered using AeroSynth.

mode will be helpful for training image classification and outlier detection models. In 'capture sky' mode, AeroSynth creates wide-field images (see Fig. 6) of the sky containing one or more objects, with each object representing a single training sample. In other words, with a single image, we can produce numerous training samples, reducing the number of synthetic images required while also saving the time required for rendering. Images generated with this mode can be used to train models for both object detection and classification.

4.3. *Object detection and classification*

Object detection and classification will be performed on images obtained from the Dalek, Alcor, and Beacon PTZ cameras (separately, the Beacon PTZ will perform outlier detection; see Sec. 4.5). The purpose will be to detect all aerial objects and then attempt to classify them. If an object is classified, with high confidence (e.g. $\geq 65\%$), as being something known, the system will be updated to stop tracking that particular object. This will be important so as to prevent the Beacon PTZ from wasting valuable time by orientating towards and tracking uninteresting targets — potentially missing objects that are truly anomalous.

To achieve this, we will utilize the YOLOv5, which is an open-source and state-of-the-art deep learning object detection and classification model (Thuan, 2021). Note that at the time of writing, YOLOv5 is the latest version, though we are aware that YOLOv8 will be available within a few months, which we will evaluate once available. YOLOv5 is capable of detecting objects in real-time video and images with high accuracy and speed using a single-stage detection pipeline that predicts bounding boxes and class probabilities for each object within an image. It operates on a feature pyramid network that can detect objects at different scales and resolutions. One of the key improvements of YOLOv5 (c.f. previous versions) is its transformer-based architecture, which enhances the model's ability to recognize objects in complex scenes by capturing contextual information (Zhu *et al.*, 2021). Moreover, YOLOv5 includes several other advanced features such as multi-scale predictions, anchor-based and anchor-free training, and improved training strategies, which contribute to its high accuracy and efficiency (Jocher, 2020).

Ultralytics (Ultralytics, 2023), the company behind YOLOv5, provides several pre-trained models. However, these pre-trained models were trained using images of people, cellphones, cars, etc., making them ideal for typical use-cases such as autonomous driving, pedestrian or traffic counting, but not well-suited to detecting and classifying objects in the sky. Therefore, we will require custom-trained models. That is, we will need to retrain the models using images of objects such as birds, kites, airplanes, clouds, etc. Ultralytics recommends training models on at least 1500 images per class, with at $\approx$10,000 instances/samples per class, including 10% of background images as a means to reduce false positives. Additionally, Ultralytics recommends that models be trained on images with similar resolutions to the images on which they will operate.

To generate the required training data, we will collect 3D models of various categories of aerial objects such as airplanes, light airplanes, helicopters, drones, military drones, birds, balloons, hot-air balloons, and blimps. We will then use AeroSynth (see Sec. 4.2) to produce four synthetic datasets, one for each camera type (see Sec. 2). After producing the synthetic data, we will retrain and evaluate each pre-trained YOLOv5 model to determine which models perform best in terms of accuracy, precision, recall, speed, and resource usage. Once appropriate models have been built, they will be integrated into the pipeline described in Sec. 4.

Given that the initial models will be trained entirely on synthetic data, we will need to collect a suitable dataset both for validating our models, as well as to collect real-world data on which subsequent models can be trained. To do this, we will use the initial models as a means to automatically label real-world data (i.e. from our cameras). Specifically, code will be written to process real-world video data using one of our custom models, and in doing so, extract detections and corresponding images from the video–the results of which will form the labeled training set. To minimize data contamination (e.g. false positives), we will set the minimum confidence threshold of the models to at least 80%. The labeled real-world dataset (in combination with synthetic data) will then be used to retrain and validate our models for improved accuracy and precision. This process will repeat until we are satisfied with the models performance. Once adequately trained, the models will replace the ones trained on synthetic data. Note that due to the ability for the YOLOv5

model to generalize well to unseen data, the risk of training bias propagating to subsequent datasets is low.

4.4. *Multi-object tracking*

One of the significant limitations of YOLO and similar object detection models is that they operate on individual frames, treating detections made in one frame as entirely distinct from those made in the next. This means that for a single frame, YOLOv5 may produce multiple detections. This presents a problem, since the Beacon PTZ can only track a single object at a time and for this reason the potential targets must be prioritized. In addition to filtering out known objects (i.e. via classification), we can reduce the number of otherwise distinct targets by tracking objects across frames. This is essentially a data association problem, which involves finding correspondences between object detections or features in the current frame and those in the previous frame, despite occlusions, noise, and other confounding factors. To address this challenge, we will employ Multi-Object Tracking (MOT) algorithms, the goal of which is to accurately track each object in the video sequence (Luo *et al.*, 2021). Fortunately, several MOT solutions exist, which we now describe.

DeepSORT (Wojke *et al.*, 2017) combines deep learning and traditional computer vision techniques to track objects in video streams. DeepSORT employs a deep neural network to extract features from object detections, which are then used to associate detections across frames and maintain tracks for each object. It also includes a track refinement step that uses appearance information to update the state of each track and correct for tracking errors. While DeepSORT shows promise, it has higher computational overheads than other methods, which may be problematic for edge-based computing where resources are constrained.

ByteTrack (Zhang *et al.*, 2022) is a multi-object tracking algorithm that uses a single-stage anchor-free detector to detect objects in each frame and a lightweight tracker that operates on the detected objects to link them across frames by using a combination of a Siamese network and a tracker network. ByteTrack achieves real-time performance and state-of-the-art accuracy on several tracking benchmarks.

Bot-SORT (Bounding Box Object Tracker with Self-Organizing Regression Trees) (Aharon *et al.*, 2022) is a tracking algorithm that uses a self-organizing regression tree to learn to predict the motion of an object in each frame. Bot-SORT also incorporates appearance information to refine the predicted motion and update the object's state.

StrongSORT (Strongly-Coupled Object Tracking) (Du *et al.*, 2022) is a tracking algorithm that uses a combination of appearance and motion cues to track objects across frames. StrongSORT models the appearance of each object as a set of deep features and uses a Kalman filter to model the object's motion. The appearance and motion models are coupled together to provide robust tracking.

OC-SORT (Online and Cascaded SORT) (Cao *et al.*, 2022) is a real-time object tracking algorithm that is designed to handle occlusions and other challenges that arise in crowded scenes. OC-SORT uses a cascaded architecture to handle occlusions, where the tracker first attempts to track each object individually and then resolves occlusions by grouping objects based on their appearance and motion information. OC-SORT also incorporates a dynamic model to adapt to changes in the scene over time.

Simple Online Real-time Tracking (SORT) (Bewley *et al.*, 2016) is a popular algorithm used for object tracking in video streams. SORT uses a combination of a Kalman filter and the Hungarian algorithm to associate object detections across multiple frames and maintain tracks for each object. The algorithm first initializes tracks for each object detected in the first frame, and then predicts the positions of those objects in subsequent frames using the Kalman filter. SORT then uses the Hungarian algorithm to associate the predicted positions with the actual object detections in the current frame. Once the associations are made, SORT updates the track for each object with its current position and other relevant information, such as velocity and size. The algorithm also handles cases where new objects appear or existing objects disappear by creating or terminating tracks as necessary. SORT is computationally efficient and can handle real-time tracking of multiple objects in video streams. However, it may suffer from issues such as ID switches, occlusion handling, and tracking errors in the presence of motion blur (Mallick, 2022).

In summary, YOLO and other object detection models operate on individual frames, and are therefore unable to distinguish between individual objects. MOT frameworks such as DeepSORT, ByteTrack, Bot-SORT, StrongSORT, OC-SORT, and SORT can be used to link objects across frames, thereby reducing the number of detections that need to be managed. Further, a byproduct of MOT is that it generates 2D trajectories for each tracked object. Such trajectories, supplemented by distance measurements obtained from triangulation using passive radar (Randall, 2023) or multiple cameras (Szenher, 2023) at a later phase of the project will provide a crucial metric with which to characterize aerial objects.

4.5. *Monitoring for outliers*

In addition to attempting to classify objects observed in images (see Sec. 4.3), the pipeline is required to estimate whether the imaged object represents something either not previously seen or very rarely seen. That is, unlike known objects such as birds, aircraft, and balloons, the properties of UAP such as shape, color, and behaviour, are poorly characterized and for this reason it is difficult to train a classification model to recognize them. Instead, we will need a model that is able to continuously adapt to the environment, learning to understand when something anomalous, such as a UAP, is detected, without being trained on what a UAP *should* look like. This mode of operation is required specifically for the analysis of images obtained by the Beacon PTZ — the high resolution targeting camera. This challenge is fundamentally an outlier detection problem, which is the process of identifying data points, events or observations which deviate significantly from those in the wider patterns underlying a data set or series. Outlier detection (sometimes called anomaly or novelty detection) is a well-studied topic, with applications in fraud and fault detection, video surveillance, insurance, safety-critical systems, and many others (Singh & Upadhyaya, 2012; Wang *et al.*, 2019; Hodge & Austin, 2004).

For the problem of detecting unusual aerial objects, we will employ an unsupervised online learning approach that will operate on the Beacon PTZ image stream. Unsupervised learning refers to a type of machine learning where an algorithm learns patterns and structures in data without the need for labeled examples or prior knowledge. That is, models are able to discover hidden relationships within the data without needing to be 'told' (i.e. presented via labeled data) what to look for. One promising approach involves the use of autoencoders.

Autoencoders (Finke *et al.*, 2021) are a type of neural network that consists of an encoder and a decoder. The encoder learns the common, high-dimensional features present in the training data[a] and compresses them to a lower-dimensional latent space, known as the bottleneck layer. In general, features should be robust to variations in illumination, scale, and rotation, and should capture the semantic content of the images. The decoder operates in reverse to the encoder. That is, the decoder uses the low-dimensional encoded representation embedded in the latent space in an attempt to reconstruct the original input image. This reconstruction is evaluated by comparing the original and reconstructed images (e.g. using Mean Squared Error, or another metric). If the 'reconstruction error' is found to be high, then the image is likely an outlier. Using this approach, our system will be able to determine when the Beacon PTZ has captured something unusual.

An important consideration is that the autoencoder model will need to adapt to the environment; it will need to learn what to consider as 'normal'. Importantly, the adaptation must be performed automatically so that little (if any) human intervention is required. For this, we can periodically fit new data to the pre-trained autoencoder and we can also update the weights of the network to improve its ability to reconstruct the new images. Automation of this 'fine tuning' of the model can be achieved using existing Automated Machine Learning (AutoML) frameworks such as Auto-PyTorch or AutoKeras (He *et al.*, 2021; Jin *et al.*, 2019).

In addition to autoencoders, we will also evaluate clustering techniques. That is, features will be extracted from a pretrained model such as an autoencoder (just described), SIFT (Lowe, 1999), HOG (Dalal & Triggs, 2005), etc. Once the features are extracted, the next step will be to identify images that are significantly different from the majority of the other images. One common approach to outlier detection is clustering, which involves grouping similar images together into

[a]We will use AeroSynth in capture-target mode to generate a set of training data.

'clusters' based on their feature vectors. The most common clustering algorithm is k-means, which iteratively assigns each image to the nearest cluster center, and updates the cluster centers based on the mean of the images in each cluster. Once the clustering is complete, outlier images can be identified as those that do not belong to any cluster or belong to a cluster with a small number of images. Although k-means requires the number of clusters to be predefined, 'online' or 'streaming' variants of the k-means algorithm can be used instead (spa, 2023; Ailon *et al.*, 2009; Wang *et al.*, 2020). This is important, since the exact number of object types (and therefore clusters) will vary depending on where our instrumentation are located.

Another approach is density estimation, which involves estimating the probability density function of the feature vectors. One common density estimation algorithm is Gaussian mixture models (GMMs) (Zong *et al.*, 2018; Acito *et al.*, 2005), which model the feature vectors as a mixture of Gaussian distributions. Outlier images can be identified as those that have low probabilities under the GMM. A third approach is to use One-Class Support Vector Machines (OCSVM's) (Amer *et al.*, 2013; Li *et al.*, 2014; Yang *et al.*, 2021). OCSVM's are a type of SVM that is trained on only one class of data (in this case, images, free from potential UAP) and can identify deviations from this norm as outliers. The algorithm fits a hyperplane (or more generally, a decision boundary) to the normal data in high-dimensional feature space. The hyperplane is selected to maximize the margin between the normal data points and the hyperplane, which helps to separate the normal data points from the potential outliers. As new data points are obtained, they are projected onto the hyperplane, and if the distance between the data point and the hyperplane is greater than a certain threshold, it is considered anomalous. In practice, the OCSVM is often implemented using a kernel function, which implicitly maps the input data into a high-dimensional feature space, where it may be easier to find a separating hyperplane, without explicitly computing the feature mapping. In addition, the kernel function is used to compute the inner products between the feature vectors, allowing for efficient computation of the decision function. This means that OCSVM's have the advantage of being computationally efficient and requiring relatively little tuning.

Regardless of the approach or approaches we choose, it must be capable of handling streaming data and of adapting to changes in the data distribution over time, and it must be computationally 'light' so as to operate on low-powered edge-computers. This means that the above methods and others will need to be explored in depth over the coming months as we develop a strategy for detecting and characterizing potential UAP.

4.6. *The decision engine*

The Decision Engine (DE) will determine which of the objects being detected by the Dalek and Alcor instruments should be targeted and tracked by the Beacon PTZ. In this initial phase of the project, we will perform a relatively simple filter-based approach. Specifically, our decision algorithm will begin by receiving detection information from YOLOv5 models operating on the Dalek and Alcor images, and will construct trajectories for each of the detected objects. The positional data of the trajectories are then mapped to world-space coordinates (Szenher, 2023) For each trajectory, the algorithm will check whether the object is classified as a known object (e.g. a bird, airplane, balloon, etc.) and has a confidence score of 80% or more, in which case the object is ignored. If the object cannot be classified or has a confidence score of less than 80%, the algorithm checks whether ADS-B data indicates the presence of an aircraft at the same coordinates. If an aircraft is present, the object is ignored. Otherwise, the object is flagged as interesting and the Beacon PTZ camera is instructed to target and track it. While tracking the object, the algorithm receives classification and confidence scores, as well as outlier scores from classification and outlier models processing operating on images obtained from the Beacon PTZ. If the object is classified as known and has a confidence score of 80% or more, or if the outlier score is low (i.e. something often seen), the object is ignored. Otherwise, if the confidence score is low or the outlier score is high, the algorithm continues to track the object. The metrics described (as well as others e.g. kinematics) will be used to compute an 'outlier score', which will be used as a means to prioritize tracked objects for tracking. Note that the 80% threshold is arbitrary and that testing will be required to determine an ideal value. See Algorithm 1 for pseudo code.

Algorithm 1. Deciding upon which objects to track.

```
while true do
    Receive detection info from YOLOv5 models
     operating on the Dalek and Alcor images
    Construct trajectories
    Map trajectories to world-space coordinates
    for each trajectory do
        if object classified as known and confidence
         score ≥ 80% then
            Flag object as uninteresting
        else if object cannot be classified or if
         confidence score is ≤ 80% then
            if ADS-B data indicates an aircraft is
             present at the same coordinates then
                Flag object as uninteresting
            else if No aircraft found at coordinates of
             detected object then
                Flag object as interesting
                Instruct the Beacon PTZ camera to
                 target the object
                Receive classification, confidence and
                 outlier scores from models operating on
                 Beacon PTZ images
                if object classified as known and
                 confidence score ≥ 80%, or if outlier
                 score is low then
                    Flag object as uninteresting
                else if confidence score is low or outlier
                 score is high then
                    Continue to track the object
                end if
            end if
        end if
    end for
end while
```

The above will form the basis of our initial decision strategy. Naturally, the DE will eventually make use of data from our other instrumentation. For instance, one approach is to combine the outputs of multiple models using a voting ensemble. Models operating on various sensor data streams could submit a 'vote' on what the detected object may represent. In this way, we would be able to validate the outputs of one or more instruments and obtain a multi-dimensional assessment of a detection. If a sufficient number of models were to agree on the classification or significance of a particular event, the appropriate action could be taken, such as ignoring or investigating further via the Beacon PTZ camera (and in later project phases, other narrow-field instruments). However, the voting approach requires that the models perform comparably well; otherwise, their inaccurate outputs may sway the system consensus, leading to errors or worse, missing a truly anomalous event. Moreover, a voting ensemble may fail to take advantage of the information embedded within an event if examined as a whole from all available sensor streams. To address these issues, Late Fusion (Akilan *et al.*, 2017) can be used, which combines the outputs of various models as inputs to a final model. That is, individual models represent informative, discriminating, and independent features; they are individual, measurable properties or characteristics of a phenomenon (Bishop, 2006). These instrument-specific, derived quantities can be used as a higher-dimensional basis on which more nuanced predictions can be made, rather than just totaling yey or nay votes. Importantly, ensemble models such as Late Fusion can often produce higher inference accuracy in comparison to individual models (Lai *et al.*, 2015). This is because errors are dealt with individually by the initial set of models and as such, they do not propagate through the fusion process and into the final 'combined' inference model. Taking a Late Fusion approach to outlier detection will therefore help minimize the rate of false positives we might otherwise observe and in turn ensure our system does not spend valuable time monitoring non-outliers. Figure 7 describes, from a high-level, the Late Fusion approach.

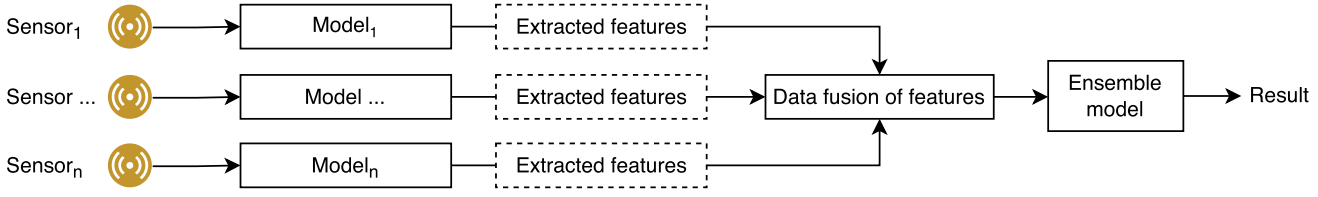

Fig. 7. (Color online) Example of late-fusion. The outputs from several models can be fused and used as input to a final decision model.

5. Offline Data Analysis

In addition to real-time analysis (described above), in which we aim to detect, characterize, and track UAP, we will also perform offline analysis. This analysis will be performed on images obtained from the various Dalek, Alcor and Beacon PTZ cameras, as well as data recorded from the other observatory sensors (see Sec. 2), including magnetometers, barometers, particle counters, temperature and humidity sensors, microphones, and more. Our goal in analyzing these data will be to both identify any unusual measurements that may have occurred simultaneously with potential UAP observations and to gain a better understanding of their characteristics. In addition, analysis will help us to validate our observations.

Offline image analysis will be primarily focused on spectral analysis (to measure reflectance, brightness, etc.), shape analysis (e.g. using segmentation to measure curvatures, area), colorimetry (to measure color distributions), and other techniques that will enable us to better understand the properties of the object captured by our cameras. Fortunately, many of the techniques required are implemented in the popular Python library, SciKit-Learn (Pedregosa *et al.*, 2011).

As mentioned, data from the observatory sensors will also be analyzed; this will be done when unusual aerial objects have been observed and tracked using the real-time system described earlier. Here, the first step will be to clean and preprocess the data, ensuring that it is accurate and complete. This will involve removing any duplicates, filling in missing values, and correcting any errors that may have been introduced e.g. by noisy sensor measurements, adverse weather conditions, etc. Any data that is not relevant to the analysis, such as sensor data collected before or after the time period of interest (the event duration), will be ignored. After cleaning the data, exploratory data analysis will be performed to gain a better understanding of the data and identify any patterns or trends. This will involve visualizing the data to identify any unusual patterns or trends, e.g. using box plots, histograms, etc. Next, we will proceed to apply various techniques for detecting outliers in our data.

5.1. *Statistical measures*

Statistical measures are commonly used for detecting outliers in data and may be particularly useful when analyzing the time series data that will be produced by instruments such as magnetometers. A popular measure is the *moving average*, where the mean value of the data within a sliding window is computed; any data point falling outside a certain range of the moving average is identified as anomalous. The *Exponentially Weighted Moving Average*, or EWMA, is another measure that emphasizes recent observations more than older ones (Zhang *et al.*, 2019). The purpose of this is to provide a more up-to-date estimate of the average value of the data within the window, which may be more sensitive to sudden changes in the data. The primary advantage of EWMA is that it provides a smoothed version of the time series that highlights recent observations while still capturing the overall trend. This makes it useful for detecting gradual changes or anomalies in the data. Another measure that can be employed is the z-score (Rousseeuw & Hubert, 2011), which is a standardized score that calculates the number of standard deviations a data point deviates from the mean of the data. This can be combined with a sliding window, such that if the z-score exceeds a predefined threshold, the data within the window can be flagged as anomalous. Naturally, the choice of the window size is a critical factor. That is, a smaller window size allows for a more detailed analysis of the data, but it may also result in more false positives. Conversely, a larger window size may miss smaller anomalies that occur within the window, but it is more likely to capture longer-lasting anomalies. Other statistical measures can be used, too, including the modified z-score, local outlier factor, interquartile range and many others; these and others will be explored.

5.2. *Clustering*

Clustering approaches can also be used and are indeed well-known for their suitability to outlier detection. Here, the basic idea is to find clusters of feature vectors, and then use the distance from these clusters to recognize outliers that may indicate an anomalous measurement. Such techniques can be useful for identifying outliers in high-dimensional data, such as data from multiple types of sensors.

k-means (Hartigan & Wong, 1979) is a widely used clustering method. It involves partitioning the data into k clusters based on their similarity, with the goal being to minimize the sum of squared distances between each data point and its nearest cluster center.

Hierarchical clustering, on the other hand, involves constructing a tree-like structure of clusters, with each data point initially assigned to its own cluster. Similar clusters are then merged together until a single cluster containing all data points is obtained.

DBSCAN (Sheridan *et al.*, 2020) is a density-based clustering algorithm and works by defining a neighborhood around each data point, and then expands the neighborhood until a minimum number of points (the 'density threshold') are found. Points that fall within a neighborhood with a density greater than the threshold are classified as 'core points'. Points that are not core points but are within the neighborhood of a core point are classified as 'border points'. Points that neither core points nor within the neighborhood of any core point are classified as 'noise points'. Once a cluster is formed, the algorithm proceeds to the next point that has not been assigned to a cluster and repeats the process until all points have been assigned to a cluster or marked as noise points.

Regardless of the particular clustering approach, the distance between clusters is typically measured using a dissimilarity metric, such as Euclidean distance (Jain *et al.*, 1999), dynamic time warping (Berndt & Clifford, 1994), or the Mahlanobis Distance (De Maesschalck *et al.*, 2000). Outliers can be identified as data points that do not fit into any of the clusters or that are far away from the nearest cluster.

5.3. *Other approaches*

Many other methods for outlier detection have been developed and described, including autoencoders (mentioned earlier), geometric models (e.g. angle-based and depth-based techniques), etc. One important approach worth mentioning is the Isolation Forrest (Liu *et al.*, 2012). Isolation Forests work by randomly selecting a feature and a split point to create a binary tree. The process is then repeated until the data points are isolated in individual 'leaves' of the 'tree'. Data points that are isolated with a small number of splits will be considered outliers. That is, outlier points are expected to require fewer splits to isolate, as they are further from the norm than typical data points. Isolation Forest is fast and scalable, can handle high-dimensional data, and does not require any assumptions about the distribution of the data. Additionally, it is capable of detecting outliers in both structured and unstructured data, and can handle both continuous and categorical variables.

In summary, offline data analysis will be performed in order to characterize detections, as well as to determine whether outlier events or unusual objects were detected by environmental sensors. The above are a few potential ways in which we will analyze our data. However, the choice of analysis technique will depend on the nature and characteristics of the data. Each technique has its advantages and disadvantages, and as such, it will be important for us to experiment so as to understand which methods are most suitable for our dataset. Fortunately, all of the above mentioned approaches (and more) are implemented and readily available via the popular Python Outlier Detection library, PyOD (Zhao *et al.*, 2019).

6. Information Security

As described in this paper, the Galileo Project system architecture consists of heterogeneous processing environments, as well as the potential to collect and produce information of national and global significance. As such, ensuring the confidentiality, integrity, and availability of system information during processing, storage, and transmission is essential to achieving the goals of the Galileo Project, which include validating hypotheses and drawing robust conclusions. The confidentiality, integrity, and availability categories of risk, commonly known as 'CIA' in cybersecurity, are widely accepted and indoctrinated across various standards, tools, and processes, such as the National Institute of Standards and Technologies (NIST) Risk Management Framework (RMF) process (Force, 2019).

The RMF process provides guidance for applying appropriate risk mitigation strategies in terms of 'security controls'. As an example, the observatory system will apply security controls for the encryption of sensitive data at rest and in transit for the primary purpose of ensuring data integrity. To ensure that a comprehensive and appropriate information security plan is in place, the observatory data processing pipeline will follow the NIST RMF process. This process includes the development and implementation of a tailored risk assessment and risk mitigation strategy, producing the System Security Plan (SSP), Security Assessment Report (SAR), and 'Plans, Assessments, and Plans of

Table 3. Security domains with risk and mitigation examples.

Security Domain	Trust Level	Description
Development	Medium	This security domain includes the environment, processes, and information for which the development, maintenance, and support of the system is dependent upon. Most security controls for this domain will be provided (inherited) via third party services and environments, though some configuration may be necessary (e.g. configure two factor authentication).
Edge Gateway	Low	This security domain includes the components of the system that support information ingress and egress. This domain also includes any user interfaces that support workflows such as system maintenance and data analysis on the edge. This domain is of highest risk, represents the most significant attack surface (low trust) as well as limited bandwidth and will require supplemental and tailored security controls.
Edge Computing	Medium	This security domain includes the components of the system that perform the edge computing services such as sensors, sensor data processing, and analytics (e.g. classification and object tracking). This domain is the second highest risk domain (medium trust) and will require supplemental and tailored security controls.
Cloud Computing	Moderate	This security domain includes the components of the system that is hosted in 3rd party cloud computing environments for post-processing and analysis. Most security controls for this domain will be inherited by 3rd party services and environments.
Site Installation Location	Low	This security domain represents the physical installation site and related physical risk and security concerns. This domain is of highest risk due to the nature of installation sites likely lacking existing physical security and security monitoring. It is antcipated that sites may vary greatly with regards to risk posture.

Action and Milestones' (POAM) artifacts. Certified and accredited information security professionals will oversee the development and execution of the RMF process. The Galileo System Security Plan will include a cyber table-top exercise that assesses the system and associated environments for unique risk and threat mitigation requirements. This exercise will leverage MITRE CAPEC (Common Attack Pattern and Enumerations) definitions and NIST security control catalog for guidance (Force, 2017).

To facilitate the risk assessment and mitigation processes, the GP system architecture has been partitioned into five distinct security domains. Each domain has specific system and process attributes that result in varying security requirements and mitigation strategies. Table 3 provides an example of an associated risk and security control (mitigation strategy) for each of the five domains of Development, Edge Gateway, Edge Computing, Cloud Computing, and Site Installation Location.

7. Conclusion

The Galileo Project is developing a multimodal, multispectral suite of instruments to detect, track and classify aerial phenomena. This paper outlines our Phase 1 computing infrastructure and system architecture, as well as our proposed real-time processing pipeline for detecting and tracking aerial outliers. In addition, we discuss promising approaches for offline data analysis and explain how we plan to ensure the security of our systems and the data we collect. Overall, our work is an important step towards an integrated computing platform for multimodal all-sky observatories designed to monitor a regional airspace for anomalous objects, and for improving data collection and analysis methods in UAP research. We hope that this paper will serve as a valuable starting point and reference for the development of future computing systems designed for investigating UAP, inspiring others to contribute to this fascinating and emerging field of research.

Acknowledgments

RC acknowledges support by the Oumuamua–Laukien fellowship awarded to the Galileo Project at Harvard University. RC would also like to express his sincerest gratitude to Daniel Llusà Ribes for his work on creating 3D models utilized by AeroSynth, Timothy Tavarez and Bradley Reimers for their contributions to software development, and to Natasha Donahue and Nicholas Gold for their help with various aspects of the GP computing

effort, as well as their discussions and feedback which have undoubtedly contributed to this work and that of Galileo Project as a whole.

References

Acito, N., Diani, M. & Corsini, G. [2005] "Gaussian mixture model based approach to anomaly detection in multi/hyperspectral images," in *Image and Signal Processing for Remote Sensing XI* (SPIE), Vol. 5982, pp. 209–217.

Aharon, N., Orfaig, R. & Bobrovsky, B.-Z. [2022] arXiv:2206.14651.

Ailon, N., Jaiswal, R. & Monteleoni, C. [2009] "Streaming k-means approximation," in *Advances in Neural Information Processing Systems*, Vol. 22 (Curran Associates).

Akilan, T., Wu, Q. J., Safaei, A. & Jiang, W. [2017] "A late fusion approach for harnessing multi-CNN model high-level features," in *2017 IEEE Int. Conf. Systems, Man, and Cybernetics (SMC)* (IEEE), pp. 566–571.

Amer, M., Goldstein, M. & Abdennadher, S. [2013] Enhancing one-class support vector machines for unsupervised anomaly detection, in *Proc. ACM SIGKDD Workshop on Outlier Detection and Description*, Association for Computing Machinery (ACM), pp. 8–15.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R. *et al.* [2001] Manifesto for agile software development, Snowbird, UT.

Bender, B. [2022] "House votes to make it easier to report UFO sightings," *Politico*, https://www.politico.com/news/2022/07/13/house-votes-easier-report-ufos-00045640.

Berndt, Donald J. & Clifford, James, [1994] "Using dynamic time warping to find patterns in time series," *AAAI Press, Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, Vol. 12, pp. 359–370, Dynamic Time Warping, Dynamic Programming, Pattern Analysis, Knowledge Discovery, Time Series, Seattle, WA, AAAIWS'94.

Bewley, A., Ge, Z., Ott, L., Ramos, F. & Upcroft, B. [2016] "Simple Online and Realtime Tracking," in *2016 IEEE Int. Conf. Image Processing (ICIP)* (IEEE), pp. 3464–3468.

Bishop, C. [2006] "*Pattern Recognition and Machine Learning, Information Science and Statistics* (Springer), https://books.google.com/books?id=qWPwnQEACAAJ.

Boikov, A., Payor, V., Savelev, R. & Kolesnikov, A. [2021] *Symmetry* **13**, 1176.

Cao, J., Weng, X., Khirodkar, R., Pang, J. & Kitani, K. [2022] arXiv:2203.14360.

Dalal, N. & Triggs, B. [2005] "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'05)* (IEEE), Vol. 1, pp. 886–893.

De Maesschalck, R., Jouan-Rimbaud, D. & Massart, D. L. [2000] "The mahalanobis distance," *Chemometrics and Intelligent Laboratory Systems*, Vol. 50, No. 1, Elsevier, pp. 1–18.

Dewi, C., Chen, R.-C., Liu, Y.-T. & Tai, S.-K. [2021] *Neural Comput. Appl.*, Springer, pp. 1–16.

Du, Y., Song, Y., Yang, B. & Zhao, Y. [2022] arXiv:2202.13514.

Finke, T., Krämer, M., Morandini, A., Mück, A. & Oleksiyuk, I. [2021] *J. High Energy Phys.* **2021**, 1.

Force, J. T. [2017] "Security and privacy controls for information systems and organizations," Technical Report, National Institute of Standards and Technology.

Force, J. T. [2019] "Risk Management Framework for Information Systems and Organizations: A System Life Cycle Approach for Security and Privacy."

Gillibrand, K. [2021] "*Gillibrand's Groundbreaking Unidentified Aerial Phenomena Amendment Included in Final NDAA*," https://www.gillibrand.senate.gov/news/press/release/gillibrands-groundbreaking-unidentified-aerial-phenomena-amendment-included-in-final-ndaa.

Happ, D., Karowski, N., Menzel, T., Handziski, V. & Wolisz, A. [2017] *Ann. Telecommun.* **72**, 41.

Hartigan, J. A. & Wong, M. A. [1979] *J. R. Stat. Soc. C* **28**, 100.

He, X., Zhao, K. & Chu, X. [2021] *Knowledge-Based Syst.* **212**, 106622.

Hittmeir, M., Ekelhart, A. & Mayer, R. [2019] "On the utility of synthetic data: An empirical evaluation on machine learning tasks," in *Proc. 14th Int. Conf. Availability, Reliability and Security*, (Association for Computing Machinery), pp. 1–6.

Hodge, V. & Austin, J. [2004] *Artif. Intell. Rev.* **22**, 85.

Jain, Anil K and Murty, M Narasimha and Flynn, Patrick J, [1999] "Data clustering: A review," *ACM Computing Surveys (CSUR)*, Vo, 31, No. 3, ACM New York, NY, USA, pp. 264–323.

Jiang, P., Ergu, D., Liu, F., Cai, Y. & Ma, B. [2022] *Procedia Comput. Sci.* **199**, 1066, doi: 10.1016/j.procs.2022.01.135.

Jin, H., Song, Q. & Hu, X. [2019] "Auto-Keras: An efficient neural architecture search system," in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining* (Association for Computing Machinery), pp. 1946–1956

Jocher, G. [2020] "YOLOv5 by Ultralytics, 7.0," doi:10.5281/zenodo.3908559.

Kang, Z., Canady, R. & Dubey, A. *et al.*, [2020] "A study of publish/subscribe middleware under different iot traffic conditions," in *Proc. Int. Workshop on Middleware and Applications for the Internet of Things* (Association for Computing Machinery), pp. 7–12, doi: 10.1145/3429881.3430109.

Kay, S. & Marple, S. [1981] "Spectrum analysis a modern perspective," in *Proc. IEEE* (IEEE), Vol. 69, p. 1380, doi: 10.1109/PROC.1981.12184.

Lai, K.-T., Liu, D., Chang, S.-F. & Chen, M.-S. [2015] *IEEE Trans. Image Process.* **24**, 2772.

Li, S., Shao, M. & Fu, Y. [2014] "Locality linear fitting one-class SVM with low-rank constraints for outlier detection," in *2014 Int. Joint Conf. Neural Networks (IJCNN)* (IEEE), pp. 676–683.

Liu, F. T., Ting, K. M. & Zhou, Z.-H. [2012] *ACM Trans. Knowl. Discovery Data* **6**, 1.

Loeb, A. [2022] arXiv:2209.02479.

Lowe, D. G. [1999] "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Computer Vision* (IEEE), Vol. 2, pp. 1150–1157.

Luo, M., Wang, K. & Cai, Z. *et al.*, [2019] *Comput. Mater. Continua* **58**, 15.

Luo, W., Xing, J. & Milan, A. *et al.*, [2021] *Artif. Intell.* **293**, 103448.

Mallick, S. [2022] "*Understanding Multiple Object Tracking using DeepSORT*," https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort.

Mead, A., Sarah, S., Paul, T., Michelle, W., Wesley, A., White, A., Galileo, P. [2023] "Multi-band acoustic monitoring of aerial signatures," *J. Astron. Instrum.* **12**(1), 2340005.

Mittal, S. [2019] *J. Syst. Archit.* **97**, 428.

ODNI. [2021] "*Preliminary Assessment: Unidentified Aerial Phenomena*," https://www.dni.gov/files/ODNI/documents/assessments/Prelimary-Assessment-UAP-20210625.pdf.

Pedregosa, F., Varoquaux, G. & Gramfort, A. *et al.*, [2011] *J. Mach. Learn. Res.* **12**, 2825.

Rakhimov, M., Mamadjanov, D. & Mukhiddinov, A. [2020] "A high-performance parallel approach to image processing in distributed computing," in *2020 IEEE 14th Int. Conf. Application of Information and Communication Technologies (AICT)* (IEEE), pp. 1–5, doi: 10.1109/AICT50176.2020.9368840.

Randall, M., Delacroix, A., Ezell, C. *et al.*, [2023] *J. Astron. Instrum.* **12**(1), 2340004.

Rousseeuw, P. J. & Hubert, M. [2011] *Wiley Interdiscip. Rev. Data Min. Knowl. Discovery* **1**, 73.

Rubio, M., Gillibrand, K. & Gallego, R. [2021] "*Rubio, Gillibrand, Gallego Applaud Inclusion of Unidentified Aerial Phenomena Amendment in National Defense bill*," https://www.rubio.senate.gov/public/index.cfm/2021/12/rubio-gillibrand-gallego-applaud-inclusion-of-unidentified-aerial-phenomena-amendment-in-national-defense-bill.

Schfer, M., Strohmeier, M., Lenders, V., Martinovic, I. & Wilhelm, M. [2014] "Bringing up OpenSky: A large-scale ADS-B sensor network for research," in *IPSN-14 Proc. 13th Int. Symp. Information Processing in Sensor Networks* (IEEE), pp. 83–94, doi: 10.1109/IPSN.2014.6846743.

Severance, C. [2012] *Computer* **45**, 6.

Sheridan, K., Puranik, T. G. & Mangortey, E. *et al.*, [2020] "An application of dbscan clustering for flight anomaly detection during the approach phase," in *AIAA Scitech 2020 Forum* (AIAA), p. 1851.

Singh, K. & Upadhyaya, S. [2012] *Int. J. Comput. Sci. Issues* **9**, 307.

Siraj, A., Loeb, A. & Moro-Martin, A., *et al.* [2022] arXiv:2211.02120.

spa [2023] "*Apache spark MLlib: Streaming k-Means*," https://spark.apache.org/docs/latest/mllib-clustering.html#streaming-k-means.

Szenher, M., Delacroix, A., Keto, E. *et al.*, [2023] *J. Astron. Instrum.* **12**(1), 2340002.

Szumski, K., Malanowski, M., Kulpa, J., Porczyk, W. & Kulpa, K. [2009] "Real-time software implementation of passive radar," in *2009 European Radar Conference (EuRAD)* (IEEE), pp. 33–36.

Thuan, D. [2021].

Ultralytics. [2023] "*Ultralytics — Revolutionizing the World of Vision AI*" https://ultralytics.com/.

Wang, H., Bah, M. J. & Hammad, M. [2019] *IEEE Access* **7**, 107964.

Wang, Z., Zhou, Y. & Li, G. [2020] "Anomaly detection by using streaming K-means and batch *K*-means," in *2020 5th IEEE Int. Conf. Big Data Analytics (ICBDA)* (IEEE), pp. 11–17.

Watters, W. A., Loeb, A., Laukien, F. *et al.*, [2023] *J. Astron. Instrum.* **12**(1), 2340006.

Wojke, N., Bewley, A. & Paulus, D. [2017] "Simple online and realtime tracking with a deep association metric," in *2017 IEEE Int. Conf. Image Processing (ICIP)* (IEEE), pp. 3645–3649.

Yang, K., Kpotufe, S. & Feamster, N. [2021] arXiv:2104.11146.

Yang, Z., Cui, Y., Li, B., Liu, Y. & Xu, Y. [2019] "Software-defined wide area network (SD-WAN): Architecture, advances and opportunities," in *2019 28th Int. Conf. Computer Communication and Networks (ICCCN)* (IEEE), pp. 1–9.

Zhang, M., Li, X. & Wang, L. [2019] *IEEE Access* **7**, 175192.

Zhang, Y., Sun, P. & Jiang, Y. *et al.*, [2022] in *Computer Vision–ECCV 2022: 17th European Conf. Proc., Part XXII*, Tel Aviv, Israel, (Springer), pp. 1–21.

Zhao, Y., Nasrullah, Z. & Li, Z. [2019] *J. Mach. Learn. Res.* **20**, 1, http://jmlr.org/papers/v20/19-011.html.

Zhu, X., Lyu, S., Wang, X. & Zhao, Q. [2021] "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," in *Proc. IEEE/CVF Int. Conf. Computer Vision* (IEEE), pp. 2778.

zmq [2021] "*2. sockets and patterns*," https://zguide.zeromq.org/docs/chapter2/.

Zong, B., Song, Q. & Min, M. R. *et al.*, [2018] "Deep auto-encoding gaussian mixture model for unsupervised anomaly detection," in *Int. Conf. Learning Representations*.