



A counter-drone visualisation platform incorporating sensor-data fusion

Lagaras, S., Amendola, D., Anderson, D.

2024

This document is a publication by the Joint Research Centre (JRC), the European Commission's science and knowledge service. It aims to provide evidence-based scientific support to the European policymaking process. The contents of this publication do not necessarily reflect the position or opinion of the European Commission. Neither the European Commission nor any person acting on behalf of the Commission is responsible for the use that might be made of this publication. For information on the methodology and quality underlying the data used in this publication for which the source is neither Eurostat nor other Commission services, users should contact the referenced source. The designations employed and the presentation of material on the maps do not imply the expression of any opinion whatsoever on the part of the European Union concerning the legal status of any country, territory, city or area or of its authorities, or concerning the delimitation of its frontiers or boundaries.

Contact information

Name: David Anderson

Address: European Commission, Joint Research Centre, Geel, Retieseweg 111, 2440, Belgium

Email: david.anderson@ec.europa.eu

Tel.: +3214571211

EU Science Hub

<https://joint-research-centre.ec.europa.eu>

JRC137155

EUR 31981 EN

PDF ISBN 978-92-68-18339-7 ISSN 1831-9424 doi:10.2760/560445 KJ-NA-31-981-EN-N

Luxembourg: Publications Office of the European Union, 2024

© European Union, 2024



The reuse policy of the European Commission documents is implemented by the Commission Decision 2011/833/EU of 12 December 2011 on the reuse of Commission documents (OJ L 330, 14.12.2011, p. 39). Unless otherwise noted, the reuse of this document is authorised under the Creative Commons Attribution 4.0 International (CC BY 4.0) licence (<https://creativecommons.org/licenses/by/4.0/>). This means that reuse is allowed provided appropriate credit is given and any changes are indicated.

For any use or reproduction of photos or other material that is not owned by the European Union permission must be sought directly from the copyright holders.

How to cite this report: European Commission, Joint Research Centre, Lagaras, S., Amendola, D. and Anderson, D., *A counter-drone visualisation platform incorporating sensor-data fusion*, Publications Office of the European Union, Luxembourg, 2024, <https://data.europa.eu/doi/10.2760/560445>, JRC137155.

Contents

Abstract	2
Foreword	3
Acknowledgements	4
1 Introduction	5
1.1 EU drones and counter-drones legislation	6
1.2 Scope of this work	8
2 C-UAS sensor-data fusion and visualisation platform	10
2.1 Direct remote identification through passive radio frequency sensor	11
2.2 Message broker	11
2.3 Data processing backend	12
2.4 In-memory database (Redis database & data storage)	15
2.5 Synthetic data generators	15
2.6 FastAPI	16
3 Sensor data fusion	18
3.1 Dynamic time warping	18
3.2 Fusing 3D trajectories using DTW	19
3.3 Python implementation of DTW	21
4 Frontend application	23
4.1 Technologies and architecture	23
4.2 Frontend frameworks and mapping libraries	24
4.3 Mapping tools in offline web applications	26
4.4 JRC DevOps Platform and GitLab repositories	27
4.5 Future developments: geo-zones and geo-fencing	27
5 Conclusions	29
References	30
List of abbreviations and definitions	32
List of figures	33
List of tables	34
Annex 1: Software, libraries, and frameworks versions	35

Abstract

This technical report describes the development of a counter-unmanned aircraft system platform for sensor-data fusion and visualisation. The report outlines the design, implementation and integration of sensor inputs for the effective detection, identification and tracking of both cooperative and non-cooperative unmanned aircraft, commonly known as drones, in airspace. With a focus on the direct remote identification sensing, the platform architecture is based on commonly used technologies, i.e., message broker, Python data processing backend, and Redis in-memory database. The objective was to develop a base platform that can be extended to deploy and test more sophisticated sensor-data fusion algorithms on real-time data. A first version of a sensor-data fusion algorithm was implemented for data from direct remote identification sensors using dynamic time warping (DTW). The Python implementation of DTW is discussed, highlighting its efficiency and applicability in real-time operations. Additionally, the frontend interface of the platform is described, offering a user-friendly graphical user interface for monitoring and controlling drone activity. The work concludes with the successful creation of a full-stack infrastructure utilizing open-source tools for data handling, processing, and visualisation. The frontend application incorporates the visualisation of the official Belgian geo-zones, and future developments are discussed, such as implementing geo-zoning and geo-fencing alerts.

Foreword

This report describes the work performed by Stelios Lagaras during a five-month scientific traineeship¹ at the European Commission's Joint Research Centre (JRC) in Geel, Belgium, between November 2023 and March 2024.

¹ More information about recruitment possibilities at JRC can be found here: <https://recruitment.jrc.ec.europa.eu>

Acknowledgements

The author would like to sincerely thank all the colleagues from the JRC's Transport and Border Security Unit for their helpful and positive attitude. The chance to work for the Joint Research Centre in such a rich and high-level scientific and diverse environment is a unique experience which can positively influence the future career of a trainee. Lastly, I would like to express my gratitude to my mentors and editors of this report, Mr Anderson and Mr Amendola – their influence was professional and supportive to this work and I hope one day we will collaborate again.

– Stelios Lagaras, March 2024

1 Introduction

Originally developed for military purposes, drones are now frequently used for disaster response, infrastructure inspection, surveillance, remote sensing, photography, recreation and more. The number of drones in use in the EU is set to grow significantly in the coming years, and they will improve greatly in terms of speed, agility, maximum range, payload capabilities, precision of sensors, and use of artificial intelligence. These developments will lead to a wider range of legitimate and lawful uses for drones.

However, drones can be used for illegal and dangerous activities such as smuggling, aviation harassment, public space harassment, cyber-attacks and terrorist attacks. In order to realise the potential of the lawful use of drones, it is necessary to address the potential threat that can be posed by non-cooperative drones. A non-cooperative drone can be classified according to the nature of non-cooperation that could include criminal, illegal (intended regulatory breach) or amateur (ignorant, careless). In recent years, plans have been uncovered involving the use of drones for terrorist attacks, and reports of malicious use of drones appear regularly in the press². There have also been sightings of suspicious drones around critical infrastructure, such as energy facilities, airports, and ports, indicating the potential misuse of drones for hostile information gathering. Drones are used by criminals engaged in smuggling across borders, or to facilitate other illicit operations, including drug trafficking and smuggling contraband into prisons (see Figure 1).

Figure 1. Early use of a UAS: a British radio-controlled aircraft from the First World War, which was the first drone to fly under control in March 1917 (left), and image of a drone for smuggling contraband into a prison in 2023 (right).



Source: left: [Imperial War Museum \(2024\)](#), right: [VRT NWS \(2023\)](#)

In this report, we adopt the term unmanned aircraft system (UAS) which includes the unmanned aircraft (UA), a ground controlling system with an operator, and the communication channel. This term was adopted by the [US Department of Defense \(2005\)](#), and it is commonly used in the technical literature and adopted in the European Single European Sky ATM Research project ([SESAR, 2015](#)).

² Examples include: (i) the plan of an inspired jihadist who was sentenced by a Spanish court in October 2022 for planning to attack a stadium during a major football match using a drone packed with explosives ([Daily Mail, 2022](#)); and (ii) a failed attempt to assassinate the Venezuelan President using explosives-laden drones ([BBC, 2018](#)).

Following the Russian invasion of Ukraine, the world has also witnessed the growing importance of UAS, which are often used in an unorthodox way, e.g. by attaching explosives, used as “kamikaze drones” or as surveillance tools. The potential of UAS to operate undetected, to be multifunctional and cost-efficient has rendered them critical weapons in multiple conflict areas. History has shown that innovations on the battlefield are subsequently translated to the civil context by criminals or terrorists. For this reason, there is currently heightened concern about the security of critical infrastructures, public spaces and borders, in terms of their vulnerability to novel attacks involving drones, and the pressing need to develop counter-UAS capability.

The Commission’s JRC is developing counter-UAS knowledge and capability via its DRONE project (European Commission, 2022) and has already published two handbooks on physical hardening (European Commission, 2023a) and protection of critical infrastructure and public spaces (European Commission, 2023b). At the time of writing, the JRC is also starting a framework contract³ for counter-drone services for research and protection of Commission infrastructure in both Geel and Brussels.

1.1 EU drones and counter-drones legislation

The European Union (EU) has adopted common rules in the field of civil aviation that include the use of UAS in European skies, aiming to establish safety and access to public flight data, awareness of the safety measures and validation control to the national authorities for individuals and different type of flying operations (European Commission, 2018). Further measures were established under Commission Delegated Regulation 2019/945 (European Commission, 2019a) and the Implementing Regulation 2019/947 (European Commission, 2019b), creating a classification of operations according to a safety-risk analysis framework.

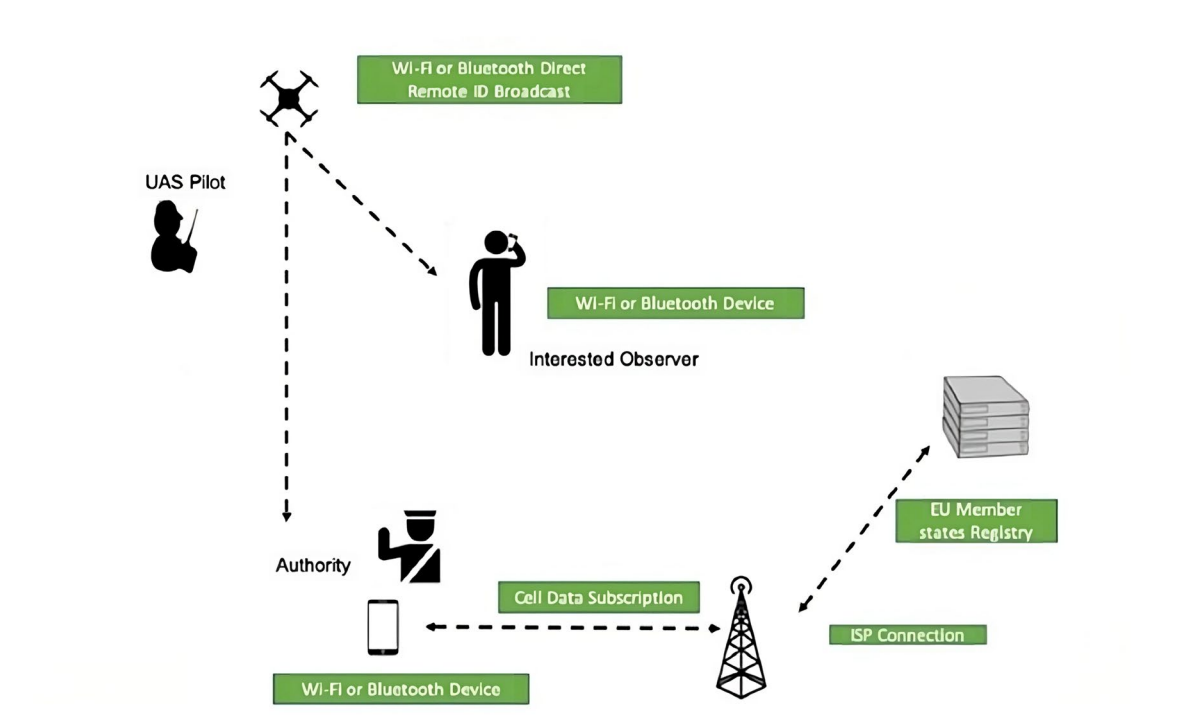
Analogous to vehicle licence plates, all unmanned aircraft (UA) above 250 g must be equipped with a direct remote identification (DRI) module and all operators should obtain an operator registration number. Pilots must obtain a pilot flying licence according to the operating class category. To this end, all UAs continuously broadcast a signal during the whole flight that contains information about the unique identification of the UA, the operator’s registration number and data about UA’s position (latitude, longitude, altitude) and other information like take-off position, brand, model, etc. (ASD-STAN, 2023). Each pilot is obliged to upload the operator’s registration number in the UA before operating, and they are accountable for proper use and following the flying rules, e.g. a drone operating on A1 category should not fly over groups of people. Sharing the operator’s registration number does not compromise privacy, because the operator’s registration number has a public and secret part – which currently only the authorities can access to get details about the operator’s real ID (ASD-STAN, 2021).

The EU has taken significant steps to address the security threats posed by drones, including incorporating drone threats in documents such as the Action Plan to support the protection of public spaces (European Commission, 2017), the EU Security Union strategy (European Commission, 2020a) and the Counter-Terrorism Agenda (European Commission, 2020b). Additionally, EU funds have been allocated to research projects on counter-UAS (European Commission, 2024).

³ <https://etendering.ted.europa.eu/cft/cft-display.html?cftId=14781>

In 2023, the Commission adopted a new Communication ([European Commission, 2023c](#)) specifically addressing the unlawful and dangerous use of civil drones. This Communication serves as a roadmap for the EU's future policy and represents a crucial step in developing a harmonised legal framework to mitigate potential risks. The Communication outlines a set of actions organised into six pillars: community building and information sharing, exploring regulatory measures, assistance in drone technology, practical guidance and operational support, research and innovation, and funding support. The Communication identifies the underpinning role of the JRC's work in a number of key actions.

Figure 2. Direct and network remote identification concept.



Source: [ASD-STAN, 2021](#)

Regulation (EU) 2019/947 defines three categories of civil drone operations: the 'open', the 'specific' and the 'certified' category. The 'open' category addresses the lower-risk civil drone operation, the category is shown in Figure 3 and it is subdivided into three subcategories, namely A1, A2 and A3. The 'specific' category addresses the riskier civil drone operation; to obtain authorisation, the drone operator is required to conduct a risk assessment and determine the requirements for the safe operation of the drone. The 'certified' category caters for the operations with the highest level of risk. Future drone flights with passengers on board such as the air taxi, for example, will fall into this category.

Regulation (EU) 2019/945 provides the classification of different drone types (classes) and flying categories for 'open' category. As we can see in Figure 3, for the open category the classification is made, mainly, according to the maximum take-off mass of the drone. Most drones sold with class identification label C1, C2, C3, C5 and C6 are already equipped with a remote identification system. All drones operating in the open category (with limited exceptions) need to be equipped with a remote identification system. In response to this European framework, most drone manufacturers are

embedding DRI at the factory, but for older drones the owners should comply by buying a remote identification module from an appropriate supplier⁴.

Figure 3. ‘Open’ category, applicable from 1 January 2024.

WHAT TYPE OF DRONE CAN I FLY?

Operation			Drone Operator / pilot			
C-Class	Max Take off mass	Subcategory	Operational restrictions	Drone Operator registration?	Remote pilot qualifications	Remote pilot minimum age
Privately build	<250g 	A1 Not over assemblies of people (can also fly in subcategory A3)	Operational restrictions on the drone's use apply (follow the QR code below)	Yes No if toy or not fitted with camera/sensor 	Read user's manual	No minimum age (certain conditions apply)
legacy < 250g						
C0						
C1	<900g 			Yes	Check out the QR code below for the necessary qualifications to fly these drones	16
C2	<4kg 	A2 Fly close to people (can also fly in subcategory A3)				
C3	<25kg 	A3 Fly far from people				
C4						
Privately build						
Legacy drones (art 20)						

**EASA**
European Union Aviation Safety Agency

#EASAdrones

together
4safety

For more details go to
<https://www.easa.europa.eu/domains/civil-drones-rpas>



Source: EASA, 2024

1.2 Scope of this work

Although DRI covers many of the simple, careless incidents, and most commercial drones will carry the DRI, a higher level of safety is needed for public spaces and critical infrastructures, where we consider malicious attackers performing more sophisticated techniques. For example, signal spoofing is a technique for mimicking a signal but with ‘synthetic’ data, e.g. a DRI packets to flood the receiver system with hundreds or thousands of ghost targets. JRC has already demonstrated the potential for a low-cost “spoofers” module that creates a huge number of false signals and can be used to overwhelm a C-UAS system and/or mask a real attack. For this reason, different kinds of sensor technologies are necessary to contribute to an effective C-UAS solution that can maximise true drones detections and minimise false alarms.

Counter-UAS solutions can utilise a combination of sensors, radars, cameras, and software algorithms to detect and track unauthorised or malicious drones in the airspace. These systems can identify the drone's location, trajectory, and other relevant information (e.g. payload type), allowing authorities to

⁴ More info: <https://www.easa.europa.eu/en/domains/civil-drones-rpas/open-category-civil-drones>

respond appropriately. In C-UAS it is important to distinguish two categories, cooperative and non-cooperative drones:

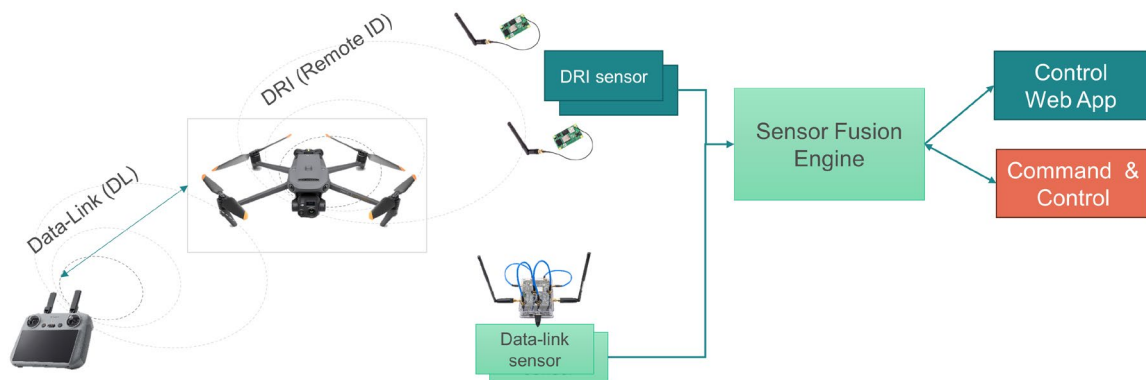
- cooperative UAS actively communicate their intentions and collaborate with infrastructure, unmanned air traffic control, and authorities;
- non-cooperative UAS do not actively contribute to their identification.

It is important that C-UAS solutions can manage both categories of UAS, e.g., by exploiting two distinct radio frequency (RF) communication channels, i.e. Remote ID broadcast messages and Data-Link channel. A C-UAS solution needs to be able to process and merge the data coming from many different RF sensors, which requires a sensor-data fusion module. Many techniques can be applied to solve the sensor-data fusion problem, e.g. deterministic algorithms or AI techniques. In this work, we will make a first version using the dynamic time warping algorithm (described later in section 3).

The addition of multiple sensor channels inevitably introduces complexity as we receive observations referring to the same object from multiple sensors, and also from the same sensor we may not have a reliable time correlation. Correctly fusing the data is not a trivial task, and a lot of experimentation is needed to resolve issues like channel synchronisation, new position (output of the fusion) approximation, choice of appropriate algorithm and much more.

In this project, we started from scratch, hence before working on the sensor data fusion algorithms we had to create the overall infrastructure to collect, manage and visualise sensor data. We needed to create a full-stack architecture, gathering the data from the sensor containing the target observation, through a process to fuse, transform and enhance the target observations, and create a graphical user interface (GUI), i.e. a web application, to visualise the information on an interactive map (see Figure 4).

Figure 4. Architecture of the solution.



Source: JRC

This report describes the tools and the architecture of the resulting C-UAS platform, including design decisions and some code snippets. The solution allows for sensor-data fusion using the dynamic time warping (DTW) algorithm, and we present some results regarding execution time of the algorithm. The process of rendering the data on the map, the challenge of serving the data on an intranet environment and the containerisation of the application services using docker are also described.

2 C-UAS sensor-data fusion and visualisation platform

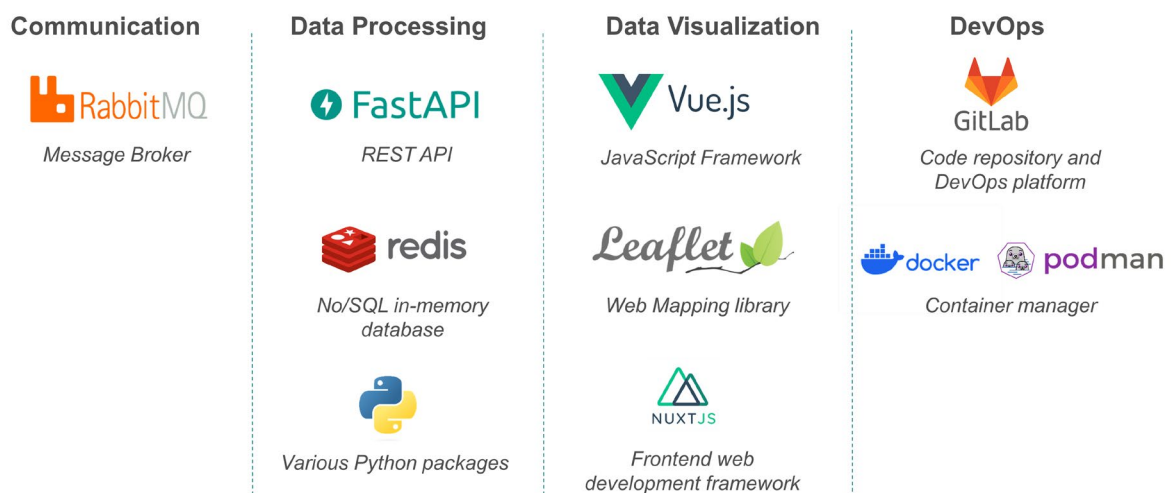
The counter-UAS sensor-data fusion and visualisation platform is a complex application, based on modern web development tools and frameworks. The software architecture is designed to be flexible and scalable, and we make use of technologies that will allow us to extend, test and maintain the platform.

We make use of a message broker, RabbitMQ, to distribute the sensor data to the backend. The backend platform is written in Python, which allows us to leverage the large number of libraries and tools already available. Data processing in the backed platform makes use of Redis, an in-memory database that makes data flow through the modules fast and convenient.

To guarantee modularity, we make use of OpenAPI standards to clearly separate backend and frontend, i.e. using FastAPI. On the frontend side, we created our web application with the NuxtJS framework, which is based on Vue.JS, and make use of components based on Leaflet for the map visualisation. A table with more details and versions of software, libraries, and frameworks is provided in Annex 1.

To enable a rapid and reliable delivery of the project code into a prototype, and maximizing the possibility to reuse the code for an extended project, we applied continuous integration and continuous deploy (CI/CD) practice. Moreover, we make use of the JRC DevOps platform⁵, in line with the JRC "Application Development support Service". This is a quite important step to improve the application's maintainability and to be able to quickly extend the platform with new functionalities. An overview of the most relevant frameworks and tools used in the project is shown in Figure 5.

Figure 5. Overview of all the main tools and frameworks used.



Source: JRC

⁵ JRC DevOps is the application development support service to streamline the software development activities of JRC scientific and administrative projects and application deployment processes in the JRC IT infrastructure. It is built on the DevOps best practices to build, test, and release software faster and more reliably.

As can be seen, a variety of different tools have been used in the different parts of the application. In the following sections, we will present some of the most relevant modules: (i) DRI sensor, (ii) message broker, i.e. RabbitMQ, (iii) data processing backend, and (iv) in-memory database, Redis.

2.1 Direct remote identification through passive radio frequency sensor

For the creation of this application, a range of different tools and technologies need to be orchestrated and integrated. Starting from the package signal, we use a commercial sensor to detect the DRI messages corresponding to cooperative drones. Different types of radio frequency (RF) sensors for C-UAS are available on the market; some RF sensors provide functionalities to track non-cooperative drones by decoding the data-link channel between UA and remote controller (RC). This kind of solution only works with specific manufactures and models. We used a sensor that is able to capture messages for Air Traffic Management (ATM) and Unmanned Traffic Management (UTM). It capture the messages and makes use of a messaging system (i.e., AMQP 0.9 or AMQP 1.0 protocols) to disseminate the information to the backend. In our solution, we make us of RabbitMQ message broker to transmit the sensor data to the backend. In the following sections, the architecture of the message broker and the necessary configuration steps to receive the observations are explained.

Figure 6. Payload message of Senhive DRI sensor (left), and photo of the sensor installed on a mast on the roof of JRC's premises in Geel, Belgium (right).

```
{
  "vehicleID": {
    "serialNumber": "1581E276CH4MR0A050W",
    "vehicleType": 2
  },
  "timestamp": "2020-06-17T08:01:01.591Z",
  "operatorID": "ABC548411-123",
  "vehicleLocation": {
    "latitude": 50.699642,
    "longitude": 15.273084,
    "accuracy": 10
  },
  "pilotLocation": {
    "latitude": 50.699642,
    "longitude": 15.273084,
    "type": 0
  },
  "altitudes": {
    "geodeticAltitude": 110.5
  },
  "trackDirection": 172,
  "groundSpeed": 2.699,
  "trackID": 5326,
  "fusion": {
    "type": 2
    "likelihood": 100
  }
}
```



Source: Senhive manual v1.02 (left), JRC (right).

2.2 Message broker

A message broker is a software that enables applications, systems, and services to communicate with each other and exchange information through messages. The main purpose of the message broker is to facilitate communication among applications in the platform, minimising the mutual awareness

between them and effectively decoupling them from each other. A message broker also validates messages, transforming if necessary and routing messages to their destination. Here we describe the setup of the RabbitMQ message broker, and introduce the basic concepts of Pika, which is a pure-Python implementation of the AMQP 0-9-1 protocol used by RabbitMQ.

Figure 7. RabbitMQ topology of queue for publisher and consumer.

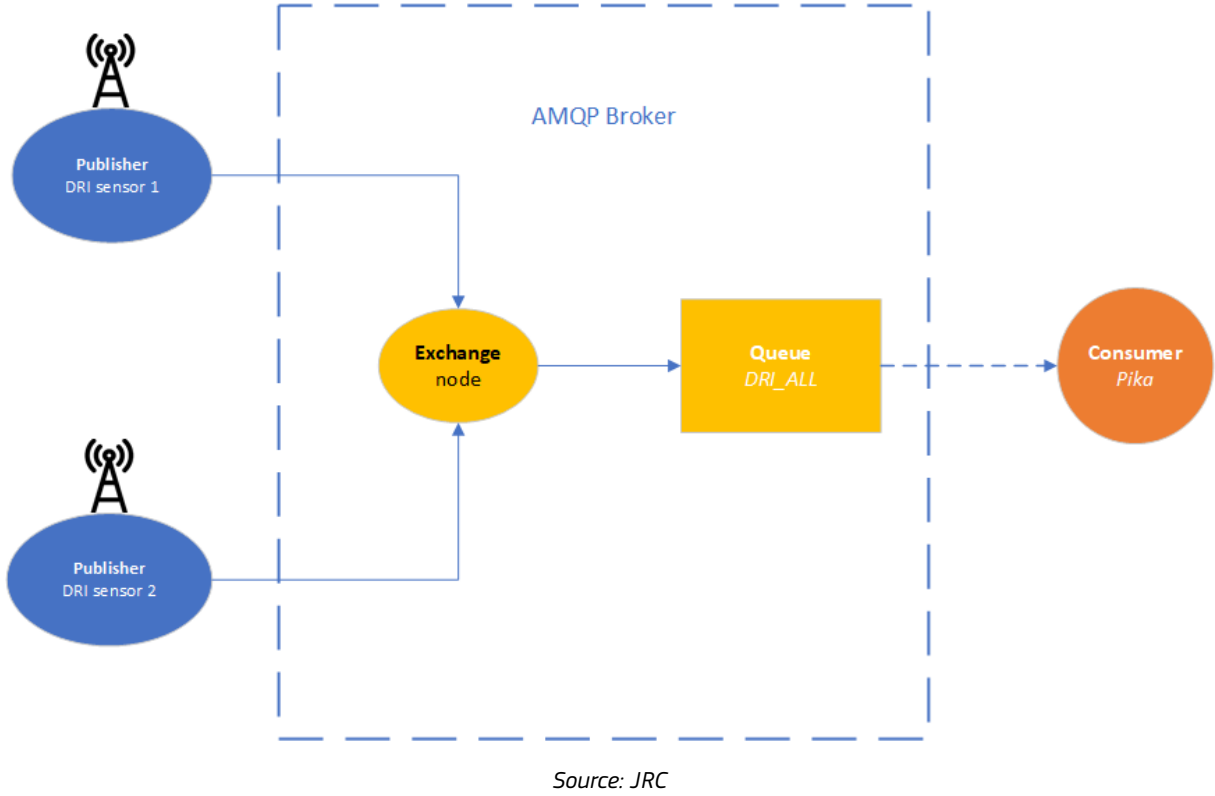


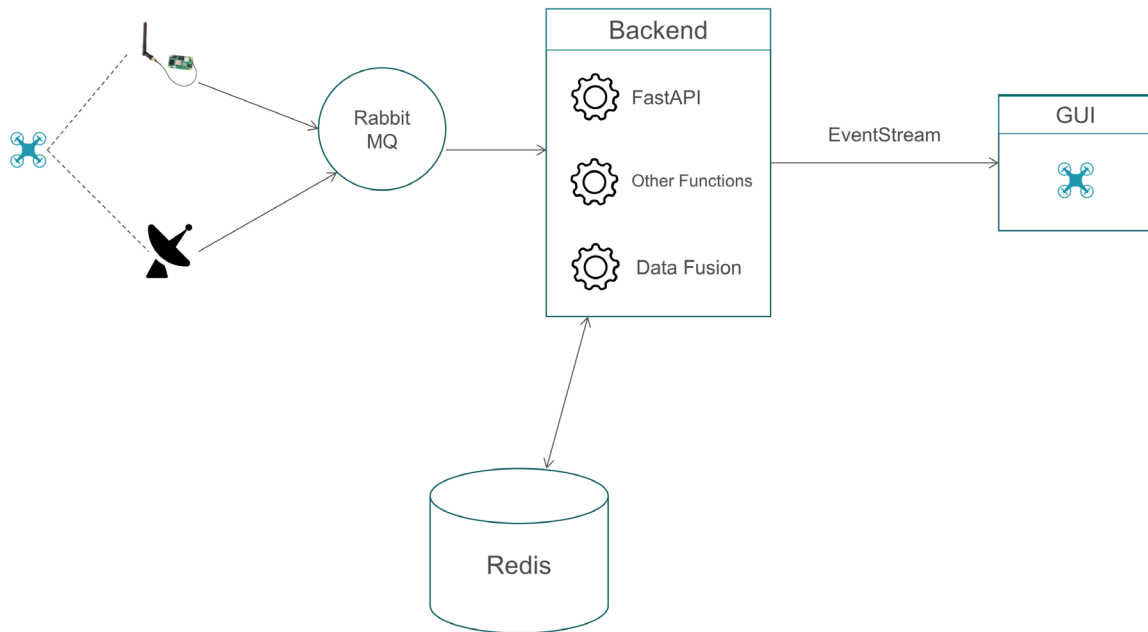
Figure 7 shows an example of the setup used in our V-LAN laboratory to test with real sensors. We have two SEN-ID DRI sensors, which are internally configured to publish the observations to a dedicated exchange that routes all the messages to one or more destination queues. In our implementation, the RabbitMQ exchange is named `sensors` and collects all the DRI sensor messages. The exchange can route to multiple queues, for instance, to better scale up. In our implementation, one queue is enough and this queue is named `DRI_ALL`. Finally the workflow is concluded with the consumer module that reads the messages from the queue. For our setup, we use an exchange with a specific type of routing to forward the messages to different queues according to their sensor name. We also make use of the convenient way to initialise the configuration, exchange types and queues of our RabbitMQ instance from the `rabbit_definitions.json` file found inside the `rabbitmq-docker` folder.

2.3 Data processing backend

In this section, the backend application and its modules are described. As introduced in previous sections, the objective was to design a modular and scalable platform. Hence our backend implements a set of APIs, designed with the OpenAPI standard, to better isolate the presentation from the processing. The decision to implement the backend application in Python is mostly due to the fact that Python is a well-known language in the scientific community, and due to the large availability of libraries and modules. Due to previous experience in Python and Node JS, a decision was taken to

use Python and the modern FastAPI web framework to prototype the APIs (see Figure 8 for an overview).

Figure 8. Abstract view of application architecture



Source: JRC

Consequently, we chose Pika as a Python for RabbitMQ message broker with a traditional blocking connection. Pika is a pure-Python implementation of the AMQP 0-9-1 protocol that tries to stay fairly independent of the underlying network support library. A blocking connection means that the thread that handles the communication will be blocked until it receives a new message from the queue. This has the trade-off from one side of simplicity, easier understanding and debugging of the application lifecycle, and on the other side the need to handle the concurrency of the threads to avoid unwanted blocking situations. A small snippet on how to connect to the RabbitMQ message broker using Pika client is shown in Figure 9.

Figure 9. Example of Python code to connect to RabbitMQ message broker

```
import pika

def consume_messages():
    """Consumer of RabbitMQ, the callback is an infinity loop triggered from
    consuming data"""

    credentials = pika.PlainCredentials(
        username=config["Credentials"]["username"],
        password=config["Credentials"]["password"],
    )
    connection = pika.BlockingConnection(
        pika.ConnectionParameters(
            host=config["broker_address"], credentials=credentials
        )
    )
    try:
        channel = connection.channel()
        channel.basic_consume(
            queue=config["queue_all"], on_message_callback=callback,
            auto_ack=True
        )
    except pika_exceptions.AMQPError as e:
        logger.error(f"Error establishing connection: {e.args}")
        return

    logger.info(" [*] Waiting for messages. To exit press CTRL+C")
    try:
        channel.start_consuming()
    except KeyboardInterrupt:
        logger.info("Interrupted")
        channel.stop_consuming()
        connection.close()

def callback(ch, method, properties, body):
    """Callback function for consuming data from RabbitMQ"""

    database.add_package_stream(r, message, sensor_name=sensor_name)
    database.add_json_datapoints(r, message, sensor_name=sensor_name)
```

Source: JRC

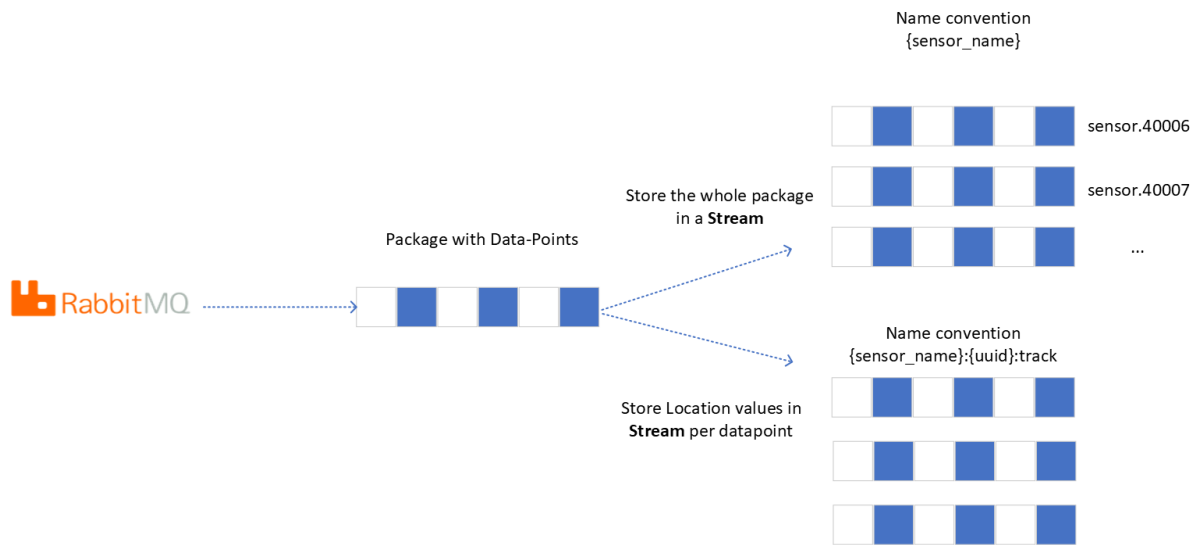
The `consume_messages()` function is responsible for the communication with RabbitMQ. When the communication is successful and a message is received, the `callback()` function is triggered and handles the new messages. For convenience, we abstract most of the function context but there is some database communication. The callback function does not use directly the received package to perform any information extraction, but interacts purely with the database in order to use historical values that are stored in the form of a time series.

One of the main ideas for future experimentation would be to try asynchronous/asyncio type of Pika adaptors, with the potential benefits of using fewer threads due to the non-blocking behaviour of concurrency. It can be expected that the more sensors/channels used in the system, the more time improvement can be obtained via asynchronous programming compared to blocking connections.

2.4 In-memory database (Redis database & data storage)

Redis was chosen as the in-memory database for managing the data coming from the sensors, due to the need for rapid prototyping, and a lack of *a priori* knowledge of different sensor-message schema. Moreover, Redis is well-known for many Python programmers, it has a good quality of documentation, and it has some useful additional features, like the possibility to make use of JSON and geospatial data formats. Finally, Redis is open-source and offers caching capabilities, which makes it a one of the most complete in-memory database solution for web applications.

Figure 10. Data points flow



Source: JRC

In Figure 10, we see the workflow of two of the main database actions. The first action, `add_package_stream`, stores each package to a Redis Stream type. Stream types are suitable for storing continuously received data like sensor data points⁶ keeping chronological order, which make them ideal for our case. The second database action, `add_json_datapoints`, creates/updates a JSON type for each data-point and a Stream to store all the location points, thus the track that will be fed to the fusion algorithm.

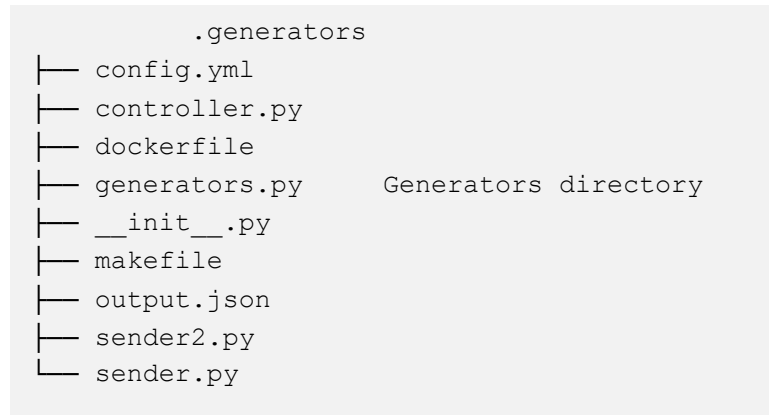
2.5 Synthetic data generators

In our first steps to reproduce synthetic sensor data broadcasted from two or more sensors, we created a synthetic data generator comprising two different scripts that randomly change the next positions of data points simulating two sensors. Later, to experiment also with synchronisation, we used our RabbitMQ instance together with a controller script to orchestrate the generators so they send together their packages synchronised with controller's period. Finally, to decouple it from our

⁶ We define data-points as a group of observations that define the message published from a sensor.

source file, a separate folder, `.generators`, was created in the backend directory that can be used as a standalone package for creating synthetic data.

Figure 11. Simulation of multiple sensors

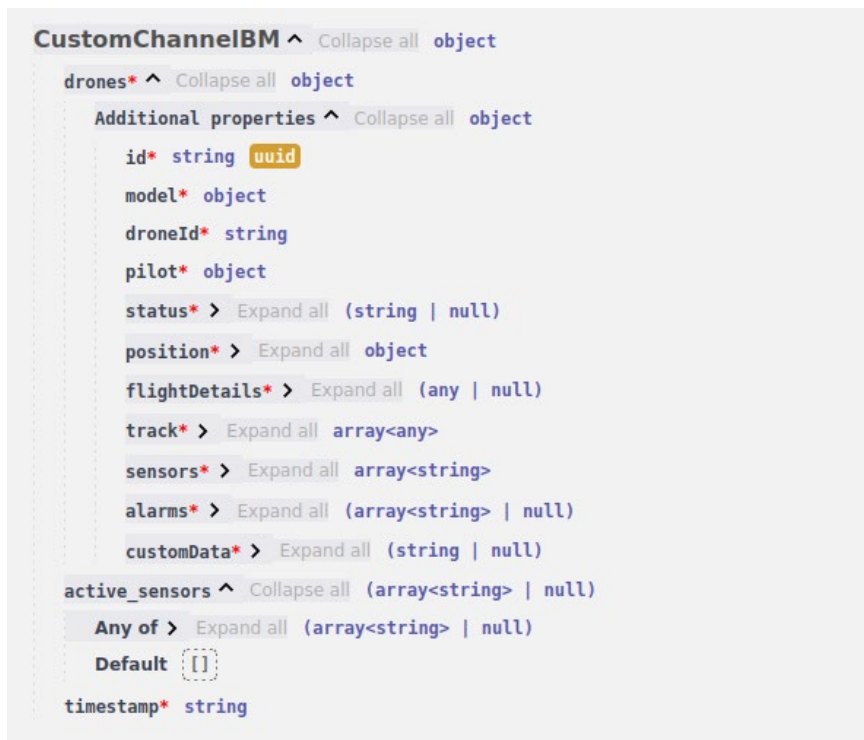


Source: JRC

2.6 FastAPI

FastAPI is a popular web framework to prototype APIs in Python, being very fast, offering a good developer experience, and has a modern and friendly documentation (unlike some older frameworks). In our application, we use it to establish an `EventSource` stream connection between the backend and the frontend of the platform. For the first implementation of the prototype, a single endpoint API is deployed (other endpoints may be developed in future). Future tasks might include, for example, get data about specific drone objects with time queries at the frontend, get the track of the drone with a certain ID in a specific timeframe. This would increase the user interaction capability of the platform, as opposed to just visualising the information.

Figure 12. Swagger details for StreamAPI: response schema of drones stream



Source: JRC

3 Sensor data fusion

Sensor fusion involves integrating data from various sensors with the aim of generating situational awareness that is more reliable and precise than what could be achieved using each sensor separately. Sensor fusion can lead to a reduction in measurement uncertainty, and can sometimes generate new insight, e.g. depth perception from two-dimensional images when fused to form a stereoscopic image. The sources used in sensor fusion can include a mix of different types of sensors, knowledge about the sensors' accuracy, historical data, as well as external knowledge and human inputs.

In C-UAS, there are different types of sensors, such as radar, acoustic sensors, electro-optical/infrared sensors and RF detectors, each with pros and cons. Adding sensors to a C-UAS system should be done judiciously, in order to ensure that the increasing complexity of the system is compensated for by an increase in performance. In its simplest form, sensor fusion can offer a universal picture of non-overlapping sensors, e.g. four radars covering a broad area in which the individual observations are merged to improving the overall monitoring process.

There are several algorithms and techniques for implementing sensor fusion e.g. Kalman filter, neural network convolution, and Gaussian processes. In this work, we describe a first step in designing a sensor-fusion process by comparing the distance between multiple trajectories of points in 3D space, in order to judge if the trajectories should be fused or not. To this end, we use dynamic time warping (DTW), which is an algorithm that can generate distance measures between sequential time series, even if the data points are not aligned or synchronised in the time domain.

3.1 Dynamic time warping

Dynamic time warping (DTW) is an algorithm for comparing two time series ([Wikipedia, 2024](#)). It produces two pieces of information: i) a mapping of which points along the two different times series correspond with each other, and ii) a measure of the similarity between the two time-series. The main characteristic of the DTW is that it can compare time series even if they are slightly delayed or stretched in the time domain. DTW was originally developed for speech recognition applications, and it has since been applied to temporal sequences of video, audio, and graphics data.

The DTW algorithm is normally implemented with the following rules:

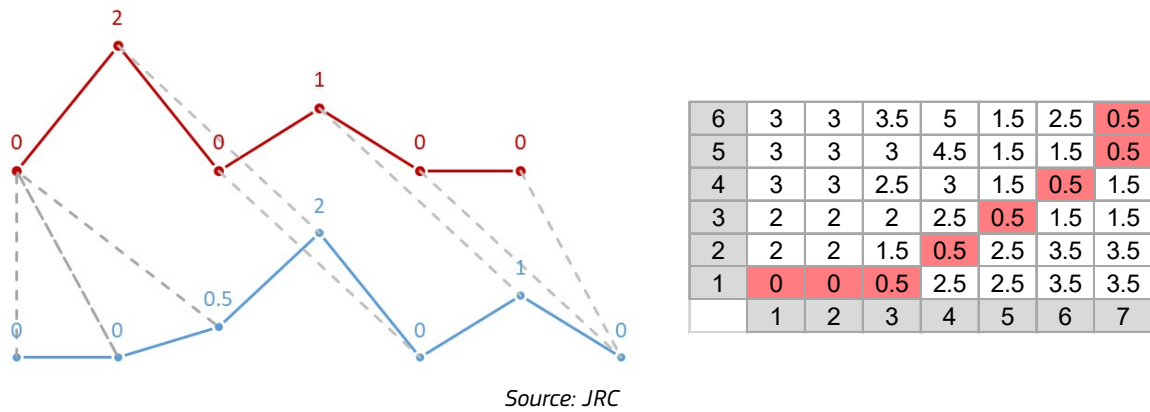
- every point from the first series must be matched with one or more from the second series, and vice versa;
- the first point from the first series must be matched with the first point from the second series (but it does not have to be its only match);
- the last point from the first series must be matched with the last point from the second series (but it does not have to be its only match);
- the mapping of the points from the first series to the second series must be monotonically increasing, and vice versa.

For two time series $x(i)$ and $y(j)$, with i in the range $(1, N)$ and j in the range $(1, M)$, the distance matrix is determined as follows:

- a $N \times M$ distance matrix is initiated;
- iterating through $i = (1, N)$ and $j = (1, M)$, the values of the distance matrix are calculated as the distance, d , between the points $d(x_i, y_j)$ plus the minimum of $[d(x_{i-1}, y_{j-1}), d(x_{i-1}, y_j), d(x_i, y_{j-1})]$, which correspond to the matching, insertion, or deletion of points, respectively;
- the mapping of points from one time series to another is obtained by starting from the upper-right corner of the matrix, $[N, M]$ and traversing back to $[0, 0]$ via the lowest of three neighbouring cells: $[i-1, j]$, $[i-1, j-1]$, or $[i, j-1]$;
- the distance measure is calculated by normalising the sum of the distances along the path by the number of points on the path.

The concept is illustrated in Figure 13. In this example, the mapping path is $[0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0, 0]$, and the normalised distance metric is therefore $3/8 = 0.375$.

Figure 13. Illustration of DTW algorithm concept. Given two time series (left), the DTW algorithm provides a distance matrix (right), which is used to generate the mapping between points and the distance metric.



3.2 Fusing 3D trajectories using DTW

In this work, we extended the DTW algorithm to three dimensions (latitude, longitude, height). The DTW distance metric was used to determine if trajectories of sensor data points were separate trajectories of multiple drones (i.e. no fusion applied), or duplicate trajectories of a single drone (i.e. sensor data should be fused). A three-phase approach was implemented:

- the original distance table is calculated which corresponds to the output of DTW for each pair of time series (i.e. 3D trajectories);
- transform the non-measurement distance unit to a similarity value where similarity $\in [0, 1]$ with 1 means exact match;

- apply a Boolean cut-off value similarity, for the example 0.9, to determine which of the time series should be fused.

When this process finishes, we are ready to merge the data series by traversing the Boolean matrix. We traverse it row by row, and we fuse the tracks by calculating their mean values. The new track object, with various useful information like which sensor data series were fused, a new unique ID, possible alarms etc. are assigned in the new drone model Pydantic schema that we have implemented (see Figure 14). The new data are stored in a special stream named `fused_stream`.

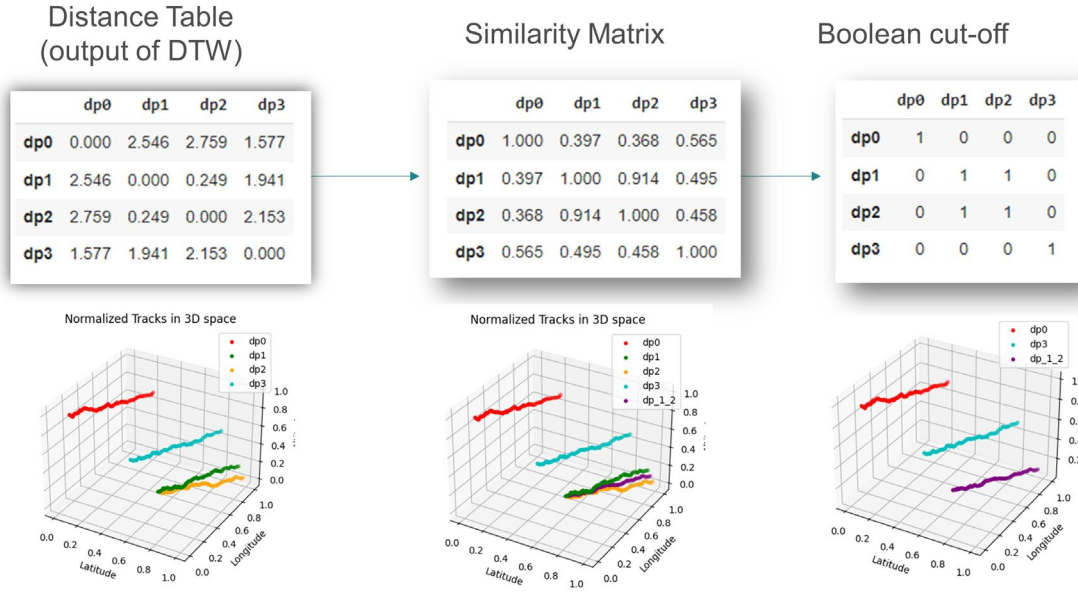
Figure 14. DroneModel Pydantic schema

```
class DroneModelBM(BaseModel):
    id: UUID
    model: dict
    droneId: str
    pilot: dict
    status: Optional[str]
    position: LocationBM
    flightDetails: Optional[object]
    track: list[object]
    sensors: list[str]
    alarms: Optional[list[str]]
    customData: Optional[str]
```

Source: JRC

An example of the 3D trajectory fusion is shown in Figure 15. On the left, we see the distance matrix in which each trajectory is compared to each other. Along the diagonal, the values are zero because the trajectories are identical. We can see that the green and yellow trajectories (dp1 & dp2) are close and have a distance of 0.249. In the middle image, the distance matrix is converted to a similarity matrix. On the right, we see the thresholded matrix, and in the graph underneath the green and yellow trajectories have been fused into a new trajectory, shown in purple.

Figure 15. Example of data fusion of 3D trajectories using dynamic time warping.



Source: JRC

3.3 Python implementation of DTW

When studying the application of DTW to the fusion of 3D trajectories, we examined the calculation time of different DTW libraries in Python. The results of the different libraries are summarised in Table 1, in terms of mean execution time and standard deviation.

Table 1. Comparison between different DTW Python libraries, results with 20 runs and time series length of 1,000 and 2,000.

DTW Python Package	Mean Execution Time	Standard Deviation
DTAIDistance distance_fast	0.0097	0.0008
TSLearn	0.0164	0.0012
PyTS	0.0421	0.0043
DTW-Python	0.0879	0.0064
FastDTW	0.6758	0.0349
Dtaidistance	3.7540	0.1216

Source: [Forecasteg.com](https://forecasteg.com/), 2024

Based on these considerations, we decided to use the `DTAIDistance` package (Meert et al., 2020) for our implementation of dynamic time warping. The `DTAIDistance` package offers several benefits, including fast C implementation with parallel mode, the ability to apply the DTW algorithm to multi-dimensional time series, a distance-to-similarity matrix conversion, and agglomerative clustering with hierarchical tree plot output.

Our algorithm reads data points received in sensor streams, and for each sensor it executes a query on the time series to get the three relevant values for the specific sensor. The length of the time series read can be configured and in our experiment (we used `DATAPOINTS_NUMBER=10`). Lastly, we create the three-dimensional location vectors for each sensor and we normalise them using the `sklearn.preprocessing` package (a submodule of `scikit-learn`). Subsequently, we have a three-dimensional matrix: i) number of drones, ii), `DATAPOINTS_NUMBER`, iii) location vector.

From Table 2, we can see that in a common scenario of less than 100 drones and a time series length of 20 with three dimensions, the execution time of the fusion is acceptable for real-time operation. There are also more operations taking place to prepare the data fusion, such as normalisation of the space and recreation of the tracks – which involves reading from the database.

Table 2. Investigation of influence of time series length on calculation time for `DTAIDistance` fast distance and similarity matrix.

Number of time series	Length of time series	Dimensions	Average time for 100 iterations	Standard Deviation
1000	100	3	23.80 s	0.0809
100	100	3	0.2634 s	0.0401
100	[50-100] ⁽¹⁾	3	0.1681	0.1586
100	20	3	0.0144 s	0.0046

⁽¹⁾ Non-homogenous length

⁽²⁾ We conducted tests to determine the scalability of the DTW libraries and algorithms in Python using the following hardware: CPU: Xeon E3-1200 v5/E3-1500 v5/6th Gen Core Processor Host.

Source: JRC

The data fusion is a CPU-bounded process, and a critical service for our application, thus we need to exhaust every possibility for optimising the execution time of the algorithm. As it is known for developers, Python by design does not allow parallelisation between different threads due to the Global Interpreter Lock. So, if we want to take advantage of more CPU resources (cores) we need to run our fusion algorithm through a separate process using the `Process` class from the `multiprocessing` package. The drawback is that we need to handle this process inside our main script `receiver.py` and use inter-process communication for triggering the fusion to start running. The solution to that, was to use the `Event` class and set the event when we start receiving some data points.

4 Frontend application

In counter-UAS, one of the most relevant activities is to command and control the operation and activities. A command and control (C2) system is designed to counter the threat posed by UAS, and it plays a crucial role in managing and coordinating C-UAS operations. It ensures effective communication, decision making, and coordination among various elements within the C-UAS architecture. In this project, it was not the intention to design a full C2 system, but rather to focus on designing a frontend application that is easy-to-use, customisable, and accessible from mobile and web.

4.1 Technologies and architecture

The frontend application should provide a visual representation of the status of the C-UAS sensor fusion platform in real time, or quasi-real time. The main purpose of this application is to give the possibility to the user to visualise the drones detected by the sensors after the processing done at the backend application. The frontend provides a visualisation of the output of the backend application, when the sensor-data fusion algorithm has consolidated the data into a list of drones. The result depends on the sensors available for the C-UAS solution, but also on the quality of the fusion algorithm and the type of targets.

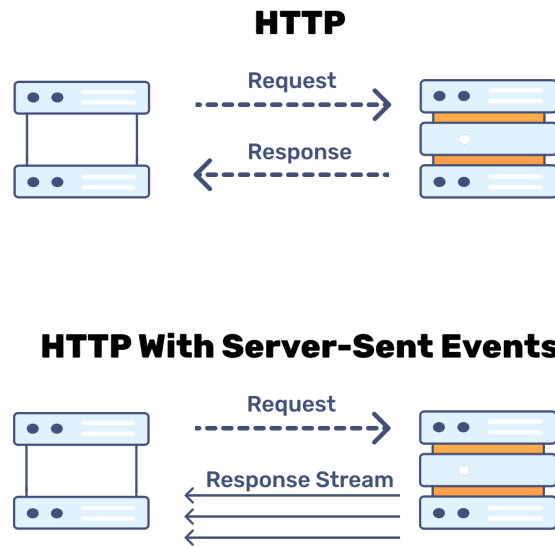
With a view to optimising the aesthetics and user experience, we used modern web frameworks (i.e. Nuxt.JS) to offer accessible, seamless, and clear interaction with the user/operator, although the application is not yet fully optimised for mobile devices. It was not the intention to design a multi-purpose GUI, nor to create a full command and control (C2) C-UAS system. Our frontend application has the advantage that it can be accessible from different types of devices, e.g. desktop, tablet, laptop, and it is easy to use and understand.

In order to have a clear separation between the backend application and the frontend application, we decided to implement an API to serve the data to the frontend and to make the interaction between the two parts of the project scalable. This approach has an additional advantage of making it possible to use other frontend applications to consume the data from the backend APIs.

Focusing on the scalability of the solution, and considering the typical HTTP request/response model, we realised that each time it was needed to get data from the backend, it was necessary to send a request and wait for the response. This model does not fit real-time concepts because of the “ping-pong” communication and the repetition of asking the same response from the API. Hence we chose a more appropriate model, namely the server-sent events (SSE), as illustrated in Figure 16.

EventSource API is a standard web API that creates an HTTP / HTTPS connection to receive push notifications from the server in the form of a stream. This communication pattern is built upon the HTTP protocol, but it is designed to enable a server to send data to the client in real time without the client needing to request it. When using SSE, the client establishes a single long-lived HTTP connection with the server using the EventSource interface. This connection uses the GET method and remains open, allowing the server to send data to the client at any time. Like the well-known WebSocket interface, which is bidirectional so the client can also send messages to the backend, SSE sends unidirectional messages (events) from server to client, better isolating the architecture and encouraging safety patterns.

Figure 16. Server-Sent Events



Source: [Bunny.net](https://bunny.net), 2024

4.2 Frontend frameworks and mapping libraries

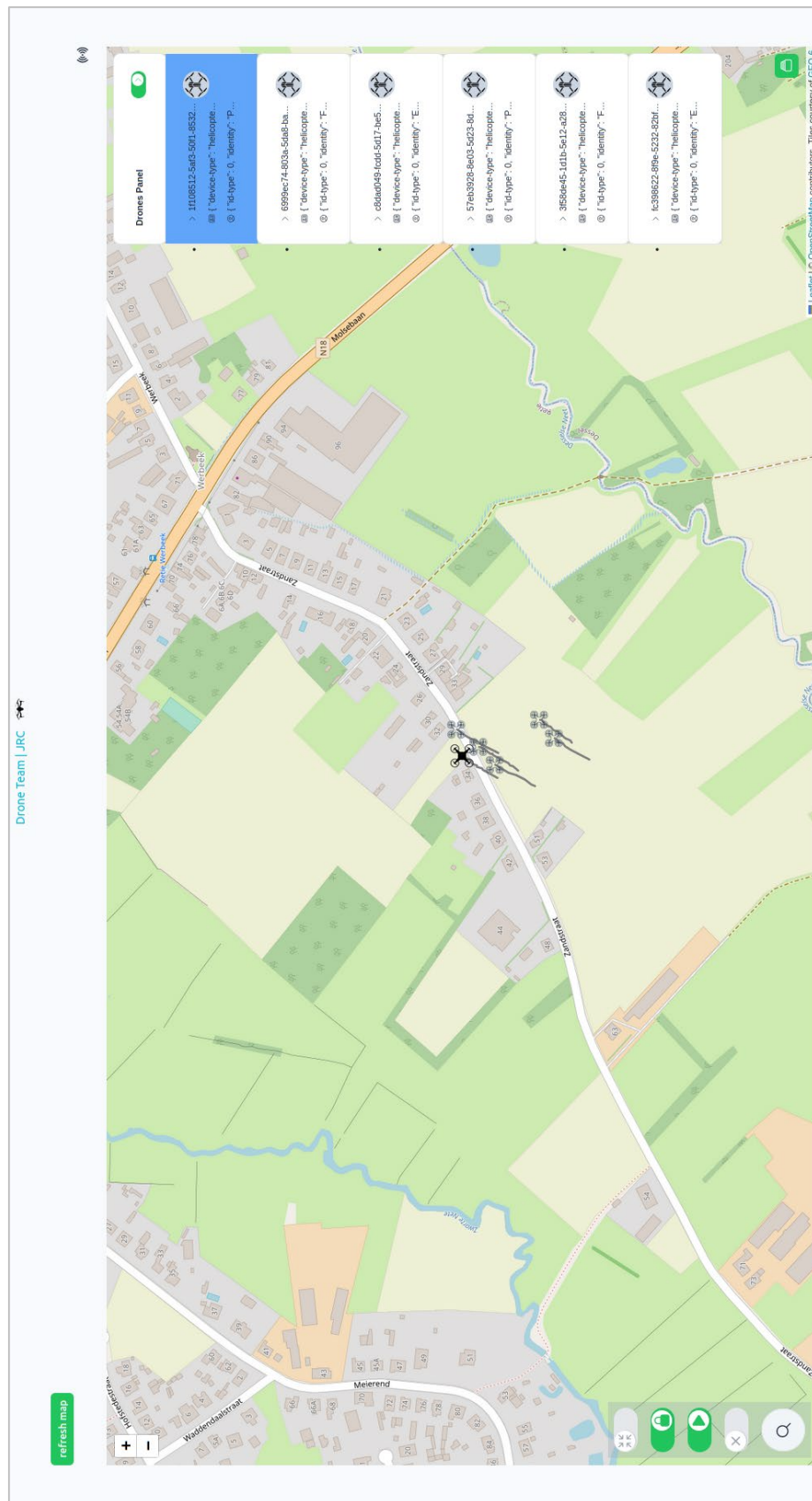
The framework chosen to implement the frontend application was Nuxt, which is a JavaScript framework for building web applications based on Vue. The version 3 of Nuxt offers a useful layer of abstraction for Vue developers, simplifying the use of concepts like routing and auto-imports of components. In this work, we used Nuxt with the Vue-Leaflet package⁷ and the Nuxt UI library⁸ for the graphical components like cards and buttons. In Nuxt UI, cards are versatile elements that can contain various types of content such as text, images, links, buttons, and other components. The map library, Leaflet, offers interactive features to help the user moving in the virtual environment, for example:

- auto-focus on specific target by clicking in a drone card;
- lock the view following a specific drone;
- highlighting the path for a specific drones;
- enable/disable the stream connection from the server;
- enable/disable multi-sensors fused data visualisation.

⁷ <https://github.com/vue-leaflet/vue-leaflet>

⁸ <https://ui.nuxt.com>

Figure 17. Frontend application; GUI comprising map with detected drones (first drone is highlighted).



Source: JRC

4.3 Mapping tools in offline web applications

C-UAS solutions are often distributed in segregated or restricted networks with limited or no internet access. Using traditional map services in offline applications can be complex and usually involves the use of external services. This was the case for our solution, which is distributed into an internal intranet without any access to the Internet. As mentioned previously, the application was developed using the JavaScript library, Leaflet⁹, for web interactive maps. Leaflet by default uses the classical raster map tiles from OpenStreetMap, which is usually a remote service, but in our cases we faced the complication to incorporate the OpenStreetMap service into our solution. OpenStreetMap provides the images (tiles) from a geographical database that are dynamically served when we navigate through the Leaflet map. To serve our own tiles offline, we use the Switch2OSM¹⁰, creating an application container with the server and the geo-database¹¹ for Belgium (deployed in our virtual machine with 8 cores CPU, 16 GB RAM).

Figure 18. Docker command to run the container with OpenStreetMap.

```
docker run \
  -p 8080:80 \
  -e THREADS=8 \
  -e "OSM2PGSQL_EXTRA_ARGS=-C 4096" \
  -v osm-data:/data/database/ \
  -d overv/openstreetmap-tile-server \
  run
```

Source: JRC

It should be noted that when the container starts it takes some time to load the bigger tiles. The server then has to generate a cache to speed up the service, hence at the beginning some slowdown is possible.

A promising strategy could be to migrate the whole map from using raster tiles to using modern vector tiles, however this kind of improvement is out of scope of this project. Instead of serving whole pictures, vector tiles serve only the geometry data of the current map view and leaves the rendering and styling of the map to the browser. Moreover, it relieves the workload of the server and lets the client's browser perform part of the process. A possible solution could be to migrate the application to a map library like MapLibre¹², allowing to use vector tiles and rendering the map directly at the client side.

⁹ <https://leafletjs.com>

¹⁰ <https://switch2osm.org>

¹¹ <https://github.com/Overv/openstreetmap-tile-server/tree/master>

¹² <https://maplibre.org>

4.4 JRC DevOps Platform and GitLab repositories

The source code of the project is stored in JRC GitLab¹³ repository, the solution architecture is based on micro-services, hence we have separated repositories for each component. In collaboration with the JRC “Application Development support Service” (JRC DevOps Platform) team, we configured the Continuous Integration/Continuous Development (CI/CD) pipeline, as established by JRC internal DevOps rules. The pipeline is based on top of three main products: 1) GitLab, 2) OpenShift and 3) Quay.

- 1) GitLab hosts all code repositories and is the JRC CI/CD solutions. We have defined a complete CI/CD workflow that covers DevOps best practices.
- 2) Openshift is the Red Hat container orchestrator platform (a flavour of Kubernetes) and hosts the applications runtimes and runs the projects’ pipelines.
- 3) Quay is the RedHat container image repository. It hosts all the container images used in the platform and it also mirrors DockerHub¹⁴ and other public repositories. All images hosted in it undergo vulnerability scans.

In the JRC DevOps Platform portfolio, there is a fourth product, Elasticsearch stack, which could be used in future version of this project for centralised logging and to store our dataset and allow us to quickly extend the application with a search engine.

The configuration of the containers and related parameters are defined in a dedicate file stored in the root of the repository, named `values.yaml`. A template of this file is provided from the JRC DevOps Platform and customised for our scope. Some of the typical field configurations are the `resources` – where the memory and CPU limits are defined, and the `ports` and `servicePorts` – for defining network level and cluster level communication respectively. Moreover, there is the default pipeline maintained by JRC DevOps Platform which is triggered every time a developer pushes a commit in a branch, making a static analysis code check, for example, it will notify if a key is exposed in your branch. Lastly, the OpenShift platform is used to deploy the applications pods.

4.5 Future developments: geo-zones and geo-fencing

In geographic information systems (GIS), *geo-zone* or *geo-fence* commonly refer to a 2D virtual perimeter or boundary around a geographic area. Geo-zones are used in various applications to trigger specific actions when a person or object enters or exits the defined area, e.g. transportation applications like taxi. A geo-zone in the U-Space refers to a 3D geographical area defined by specific coordinates where U-Space services and regulations are applied. Geo-zones are portions of airspace where UAS operations are facilitated, restricted or excluded. These geo-zones can be used for various purposes, such as defining areas for drone operations, managing air traffic, or implementing safety measures. By creating geo-zones, U-Space aims to ensure the safe and efficient integration of UAS into the airspace.

¹³ <https://gitlab.jrc.ec.europa.eu/jrc-projects/drone/sensor-fusion>

¹⁴ <https://hub.docker.com>

In our solution, we have integrated the visualisation of official geo-zones for Belgium to help the users and facilitate the C-UAS operations. The Belgian geo-zones are collected from the official service Droneguide¹⁵ from Skeyes Belgian air traffic controller. The data are downloaded from the official map in GeoJSON format.

Adding the geo-fencing functionality to have automatic alerts for unauthorised UAS entering in a specific geo-zone is left for future development. However, we already designed a partial solution by adding a specific geolocation and making use of Tile38 geo-fencing tool. The main idea is that all the cooperative UAS are broadcasting the DRI messages, the location of each UASs is cross-checked with the geo-server, and if a drone cross a restricted area an alarm is shown on the GUI. Geo-fencing can simplify and speed up the phase of *intervention* to mitigate the risk coming from drone in restricted areas.

¹⁵ <https://map.droneguide.be>

5 Conclusions

This project has successfully established a comprehensive full-stack **counter-UAS sensor-data fusion and visualisation platform** that can integrate multiple sensor inputs for effective drone monitoring within the European Union's airspace. The platform is designed to address both cooperative and non-cooperative drones, given the rapid proliferation of drone technology and its varied applications. Through the implementation of advanced technologies such as message brokers, a Python data processing backend, and Redis **in-memory database**, the platform has significant potential as a foundational system for real-time data analysis and sensor fusion.

A key innovation within the project is the designing of a **trajectory fusion** algorithm with the application of the **dynamic time warping** (DTW) algorithm to fuse 3D trajectories of sensor data, which provides robust alignment of time series that are not synchronised temporally. The Python implementation of the DTW algorithm has proven to be efficient for real-time operations and forms the basis for future exploration and development of more complex sensor fusion algorithms.

The **frontend interface**, characterised by its user-friendliness, enables operators to monitor and control drone activity effectively. It represents an accessible and scalable solution that can adapt to various devices, enhancing the operational capabilities of those tasked with airspace security.

In conclusion, the project has laid a **solid groundwork** for future advancements in C-UAS technology. With the successful integration of official Belgian geo-zones within the frontend application, the platform is well-positioned to incorporate geo-zoning and geo-fencing alerts (a backend module would be implemented for that), thereby elevating its functionality. As drone technology continues to evolve, so too will the challenges of monitoring and securing airspace. The platform developed in this project stands as a versatile tool, ready to meet these challenges through continued innovation and integration of emerging technologies.

References

- ASD-STAN (2021). *Direct Remote ID - Introduction to the European UAS Digital Remote ID Technical Standard*. Accessed 28 May 2024, Retrieved from https://cms.stan-shop.org/uploads/2024/01/ASD-STAN_DRI_Introduction_to_the_European_digital_RID_UAS_Standard.pdf
- ASD-STAN (2023). *ASD-STAN prEN 4709-002 P1 Corrigendum 1 (February 2023) Aerospace series – Unmanned Aircraft Systems – Part 002: Direct remote identification requirements*. Available at <https://stan-shop.org/en/catalog/item/75298>
- BBC (2018). Venezuela President Maduro survives 'drone assassination attempt', 5 August 2018, <https://www.bbc.com/news/world-latin-america-45073385>
- Bunny.net (2024). *What are SSE (Server-Sent Events) and how do they work?* <https://bunny.net/academy/http/what-is-sse-server-sent-events-and-how-do-they-work>
- Daily Mail (2023). ISIS fanatic who planned to bomb Barcelona-Madrid match with explosive-packed drone to atone for drinking alcohol in the past and 'assure his place in paradise' is jailed for three years, 24 October 2022, <https://www.dailymail.co.uk/news/article-11348743/ISIS-fanatic-planned-bomb-Barcelona-Madrid-match-jailed-three-years.html>
- EASA (2024). *Open Category - Low Risk - Civil Drones*, Retrieved from <https://www.easa.europa.eu/en/domains/civil-drones-rpas/open-category-civil-drones>
- European Commission (2017). Communication from the Commission To the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions – Action Plan to support the protection of public spaces, COM/2017/0612 final, 18 October 2017, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52017DC0612>
- European Commission (2018). Regulation (EU) 2018/1139 of the European Parliament and of the Council of 4 July 2018 on common rules in the field of civil aviation and establishing a European Union Aviation Safety Agency, and amending Regulations (EC) No 2111/2005, (EC) No 1008/2008, (EU) No 996/2010, (EU) No 376/2014 and Directives 2014/30/EU and 2014/53/EU of the European Parliament and of the Council, and repealing Regulations (EC) No 552/2004 and (EC) No 216/2008 of the European Parliament and of the Council and Council Regulation (EEC) No 3922/91, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32018R1139>
- European Commission (2019a). Commission Delegated Regulation (EU) 2019/945 of 12 March 2019 on unmanned aircraft systems and on third-country operators of unmanned aircraft systems, C/2019/1821, https://eur-lex.europa.eu/eli/reg_del/2019/945/oj
- European Commission (2019b). Commission Implementing Regulation (EU) 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft, C/2019/3824, https://eur-lex.europa.eu/eli/reg_impl/2019/947/oj
- European Commission (2020a). Communication from the Commission to the European Parliament, the European Council, the Council, the European Economic and Social Committee and the Committee of the Regions on the EU Security Union Strategy, COM/2020/605 final, 24 July 2020, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52020DC0605>
- European Commission (2020b). Communication from the Commission to the European Parliament, the European Council, the Council, the European Economic and Social Committee and the Committee of the Regions – A Counter-Terrorism Agenda for the EU: Anticipate, Prevent,

- Protect, Respond, COM/2020/795 final, 9 December 2020, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52020DC0795>
- European Commission (2022). Newsroom – JRC DRONE Project to Counter UAS for CI, 2 July 2022, Accessed 30 May 2024, <https://ec.europa.eu/newsroom/cipr/items/752427>
- European Commission (2023a). Karlos, V., Larcher, M., *Protection against Unmanned Aircraft Systems – Handbook on UAS risk assessment and principles for physical hardening of buildings and sites*, Publications Office of the European Union, EUR 31458 EN, <https://data.europa.eu/doi/10.2760/969680>
- European Commission (2023b). Hansen, P., Pinto Faria, R., *Protection against Unmanned Aircraft Systems – Handbook on UAS protection of critical infrastructure and public space – A five phase approach for C-UAS stakeholders*, EUR 31454 EN, Publications Office of the European Union, <https://data.europa.eu/doi/10.2760/18569>
- European Commission (2023c). Communication from the Commission to the Council and the European Parliament on countering potential threats posed by drones, COM/2023/659 final, 18 October 2023, <https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=COM:2023:659:FIN>
- European Commission (2024). Stepniak, M., Cheimariotis, I., Lodi, C., Rataj, M., Zawieska, J., Grosso, M. and Marotta, A., *Research and Innovation on Drones in Europe*, Publications Office of the European Union, Luxembourg, 2024, JRC137334, <https://dx.doi.org/10.2760/02357>
- Forecastegy.com (2024). *5 Dynamic Time Warping (DTW) Libraries in Python With Examples*, Retrieved from <https://forecastegy.com/posts/dynamic-time-warping-dtw-libraries-python-examples/#which-dtw-library-is-faster>
- Imperial War Museum (2024). *A Brief History of Drones*, Retrieved from <https://www.iwm.org.uk/history/a-brief-history-of-drones>
- Meert, W., Hendrickx, K., Van Craenendonck, T., Robberechts, P., Blockeel, H., & Davis, J. (2020). *DTAIDistance (v2.3.10)*. Zenodo. <https://doi.org/10.5281/zenodo.7158824>
- RabbitMQ (2024). *RabbitMQ tutorial*. Retrieved from: <https://www.rabbitmq.com/tutorials/tutorial-five-python.html>
- SESAR (2015). *European ATM master plan – The roadmap for delivering high performing aviation for Europe*, Edition 2015, Publications Office, 2015, <https://data.europa.eu/doi/10.2829/240873>
- US Department of Defense (2005). *Unmanned Aircraft Systems Roadmap 2005-2030*, Retrieved from <https://rosap.ntl.bts.gov/view/dot/18248>
- VRT NEWS (2023). Criminal gang that used drones to supply French and Belgian prisons with drugs and cigarettes busted, 8 December 2023, Accessed May 2024, Retrieved from <https://www.vrt.be/vrtnws/en/2023/12/08/criminal-gang-that-used-drones-to-supply-french-and-belgian-pris/>
- Wikipedia (2024). Dynamic time warping, Accessed May 2024, Retrieved from https://en.wikipedia.org/wiki/Dynamic_time_warping

List of abbreviations and definitions

AI	artificial intelligence
AMQP	advance message queuing protocol
API	application programming interface
ATM	air traffic management
CI/CD	continues integration / continues delivery
C-UAS	counter unmanned aircraft system
CPU	central processing unit
DRI	direct remote identification
DTW	dynamic time warping
EASA	European Aviation Safety Agency
GIS	geographic information system
GUI	graphical user interface
JRC	Joint Research Centre
JSON	JavaScript Object Notation
RF	radio frequency
RFID	radio frequency identification
RID	remote identification
SESAR	Single European Sky ATM Research
SSE	server-sent events
UA	unmanned aircraft
UAS	unmanned aircraft system
UTM	unmanned traffic management
VLAN	virtual local area network

List of figures

Figure 1. Early use of a UAS: a British radio-controlled aircraft from the First World War, which was the first drone to fly under control in March 1917 (left), and image of a drone for smuggling contraband into a prison in 2023 (right).....	5
Figure 2. Direct and network remote identification concept.....	7
Figure 3. ‘Open’ category, applicable from 1 January 2024.....	8
Figure 4. Architecture of the solution.....	9
Figure 5. Overview of all the main tools and frameworks used.....	10
Figure 6. Payload message of Senhive DRI sensor (left), and photo of the sensor installed on a mast on the roof of JRC’s premises in Geel, Belgium (right).....	11
Figure 7. RabbitMQ topology of queue for publisher and consumer.	12
Figure 8. Abstract view of application architecture.....	13
Figure 9. Example of Python code to connect to RabbitMQ message broker	14
Figure 10. Data points flow	15
Figure 11. Simulation of multiple sensors	16
Figure 12. Swagger details for StreamAPI: response schema of drones stream.....	17
Figure 13. Illustration of DTW algorithm concept. Given two time series (left), the DTW algorithm provides a distance matrix (right), which is used to generate the mapping between points and the distance metric.....	19
Figure 14. DroneModel Pydantic schema	20
Figure 15. Example of data fusion of 3D trajectories using dynamic time warping.....	21
Figure 16. Server-Sent Events	24
Figure 17. Frontend application; GUI comprising map with detected drones (first drone is highlighted).....	25
Figure 18. Docker command to run the container with OpenStreetMap.	26

List of tables

Table 1. Comparison between different DTW Python libraries, results with 20 runs and time series length of 1,000 and 2,000.....	21
Table 2. Investigation of influence of time series length on calculation time for <code>DTAIDistance</code> fast distance and similarity matrix.....	22
Table 3. List of software versions.....	35

Annex 1: Software, libraries, and frameworks versions

In the table below, we list the main version for most relevant software, libraries and frameworks used to build the prototype platform.

Table 3. List of software versions.

Software	Library	Version
Python		3.11
Python Modules	fastapi	0.109.0 or above
Python Modules	redis	5.0.0 or above
Python Modules	uvicorn	0.27.0.post1 or above
Python Modules	sse-starlette	2.0.0 or above
Python Modules	schedule	1.2.1 or above
Python Modules	matplotlib	3.8.2 or above
Python Modules	dtaiDistance	2.3.11 or above
Python Modules	pika	1.3.2 or above
RabbitMQ		3.11.13
Redis		7.2.0
NodeJS		20
NodeJS dependencies	@heroicons/vue	2.0.18 or above
NodeJS dependencies	@nuxt/ui	2.10.0 or above
NodeJS dependencies	@vueuse/nuxt	10.6.1 or above
NodeJS dependencies	leaflet	1.9.4 or above

Source: JRC

Getting in touch with the EU

In person

All over the European Union there are hundreds of Europe Direct centres. You can find the address of the centre nearest you online (european-union.europa.eu/contact-eu/meet-us_en).

On the phone or in writing

Europe Direct is a service that answers your questions about the European Union. You can contact this service:

- by freephone: 00 800 6 7 8 9 10 11 (certain operators may charge for these calls),
- at the following standard number: +32 22999696,
- via the following form: european-union.europa.eu/contact-eu/write-us_en.

Finding information about the EU

Online

Information about the European Union in all the official languages of the EU is available on the Europa website (european-union.europa.eu).

EU publications

You can view or order EU publications at op.europa.eu/en/publications. Multiple copies of free publications can be obtained by contacting Europe Direct or your local documentation centre (european-union.europa.eu/contact-eu/meet-us_en).

EU law and related documents

For access to legal information from the EU, including all EU law since 1951 in all the official language versions, go to EUR-Lex (eur-lex.europa.eu).

EU open data

The portal data.europa.eu provides access to open datasets from the EU institutions, bodies and agencies. These can be downloaded and reused for free, for both commercial and non-commercial purposes. The portal also provides access to a wealth of datasets from European countries.

Science for policy

The Joint Research Centre (JRC) provides independent, evidence-based knowledge and science, supporting EU policies to positively impact society



EU Science Hub

joint-research-centre.ec.europa.eu



Publications Office
of the European Union