SUNRISE

Strategies and Technologies for **Un**ited and **R**esilient Critical **I**nfrastructures and Vital **S**ervices in Pandemic-Stricken **E**urope

D7.3 Infrastructure inspection tool and training guide V2

Document Identification			
Status	Final	**Due Date**	31/05/2024
Version	1.0	**Submission Date**	30/05/2024

Related WP	WP7	**Document Reference**	D7.3
Related Deliverable(s)	D7.1, D7.2	**Dissemination Level (*)**	PU
Lead Participant	SKY	**Lead Author**	Tasos Gkamaris
Contributors	ATS, XLAB, SKYLD, ICS, MZI, ELS, SZ, SZI, PIL, ACO, INS, HDE, TLF, TS, HER, CCL, INT	**Reviewers**	IMA SQD

Keywords:

Critical infrastructure, remote inspection, artificial intelligence, satellite imagery, UAV

(*) Dissemination level: **(PU)** Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page). **(SEN)** Sensitive, limited under the conditions of the Grant Agreement. **(Classified EU-R)** EU RESTRICTED under the Commission Decision No2015/444. **(Classified EU-C)** EU CONFIDENTIAL under the Commission Decision No2015/444. **(Classified EU-S)** EU SECRET under the Commission Decision No2015/444.

Document Information

List of Contributors	
Name	Partner
Tasos Gkamaris	SKY
Romeo Bratska	SKY
Ismini Rousounidou	SKY
Yiannis Arvanitakis	SKY
Enea Qerama	SKY
Andreja Markun	TS
Mario Triviño	ATS
Daniel Vladušič	XLAB
Anja Zdovc	XLAB
Urban Bavčar	ELS
Josip Radman	MZI
Andrea Bello	HDE
Lars Ake Olofsson	ACO

Document History			
Version	Date	Change editors	Changes
0.1	12/02/2024	Tasos Gkamaris (SKY)	Table of Content included, and first round of general document general data.
0.2	25/03/2024	Tasos Gkamaris (SKY)	First round of content in Sections: 1 (1.1, 1.2, 1.3), 2 (2.2), 3 (3.1, 3.2, 3.3, 3.5, 3.6), 4, 5
0.3	03/04/2024	Tasos Gkamaris (SKY)	Second round of content in Sections: 1 (1.1, 1.2, 1.3), 2 (2.2), 3 (3.1, 3.2, 3.3, 3.5, 3.6), 5
0.4	14/04/24	Tasos Gkamaris (SKY)	Final round of content in all sections and first full draft integration
0.5	20/04/2024	Tasos Gkamaris (SKY)	Full draft integration
0.5.1	26/04/2024	Daniel Vladušič (XLAB)	Final draft contribution Section 2
0.5.2	26/04/2024	Mario Triviño (ATS)	Final draft contribution Section 3
0.5.3	26/04/2024	Urban Bavčar (ELES)	Final draft contribution Section 5
0.6	28/04/2024	Tasos Gkamaris (SKY)	Final draft contribution Section 4
0.61	30/04/2024	Mario Triviño (ATS)	Final draft contribution Section 3
0.7	30/04/2024	Tasos Gkamaris (SKY)	Final draft integration
0.8	05/05/2024	Andrea Bello (HDE)	Final draft contribution Section 5
0.9	06/05/2024	Tasos Gkamaris (SKY)	Final draft integration
0.91	15/05/2024	Tasos Gkamaris (SKY)	Changes on references index after 1st revision
0.92	30/05/2024	Juan Alonso (ATS)	Quality Assessment
1.0	30/05/2024	Aljosa Pasic (ATS)	Final version

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Tasos Gkamaris (SKY)	29/05/2024
Quality manager	Juan Andrés Alonso (ATS)	30/05/2024
Project Coordinator	Aljosa Pasic (ATS)	30/05/2024

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
2FA	Two-Factor Authentication
AI	Artificial Intelligence
BVLOS	Beyond Visual Line Of Sight
CI	Critical Infrastructure
CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neural Network
ESA	European Space Agency
EU	European Union
FN	False Negative
FP	False Positive
FTPS	File Transfer Protocol Secure
FPV	First Person View
GDPR	General Data Protection Regulation
GPT	Generative Pre-trained Transformer
GUI	Graphical User Interface
IoU	Intersection over Union
LEVIR	LEarning, VIsion and Remote
LLM	Large Language Model
LSI	Legacy Systems Integration
MAE	Mean Square Error
mAP	mean Average Precision
MLLM	Multimodal Large Language Model
MQTT	Message Queuing Telemetry Transport
MSBC	Multisource build-up change
MTOW	Maximum Take Off Weight
NeRF	Neural Radiance Fields
POI	Point of Interest
REST API	Representational State Transfer Application Programming Interface
SAM	Segment Anything Model
SAM-HQ	Segment Anything Model in High Quality
TP	True Positive
UAS	Unmanned Aircraft System
UAV	Unmanned Aerial Vehicle
UI	User Interface
VHM	Vegetation Height Model
VLM	Visual-Language Model
VLOS	Visual Line Of Sight
VQA	Visual Question Answering
WP	Work Package
YOLO	You Only Look Once

Executive Summary

The SUNRISE tool aims to provide users with real-time information on critical infrastructure by analyzing image and video feeds. This information includes details on damaged components, structural issues, corrosion, and obstructing vegetation. The tool utilizes AI-assisted components to process data from satellite and UAV feeds and present it in the user interface.

This document focuses on the AI mechanisms enhancing data for the SUNRISE tool and outlines steps taken to integrate satellite and UAV inspection tools, develop the dashboard UI, and connect with legacy CI systems for improved functionality. It is a technical report wherein establishes a robust foundation upon which further development can ensue, incorporating additional features and models to enhance usability and address as many inspection challenges as possible.

This D7.3 introduces the second iteration in the process of implementing remote inspection tools for critical infrastructures. Within the scope of WP7, it represents the updated efforts to implement the concepts introduced in D7.2[42], aiming to address all the requirements of CI stakeholders outlined in D3.2. The progression of the content herein will be evident in subsequent iterations (D7.4-D7.6).

Experiments in remote inspection methods with satellite images were performed using different methods in order to have a wider variety of suitable methods prepared for piloting (Pilot 1). Additional high-resolution images were obtained to evaluate (with CIs) the usefulness of the developed methods.

This document captures significant technological progress since the previous deliverable, illustrating a notable enhancement in remote inspection tools. It highlights the integration of state-of-the-art models, including the implementation of new Visual-Language Models V-LLM and disruptive technologies for real-time open-vocabulary detection. These advancements are implemented within the UAV image analytics platform, enhancing the capability to process and analyze visual data in dynamic scenarios effectively and efficiently. The results obtained reveal great potential for the application of these technologies, marking a significant step forward in developing a broad-spectrum zero-shot UAV inspection tool.

This deliverable outlines the coordination efforts for executing Pilot 1 in critical infrastructure fields. The key information is outlined in Section 5. By running targeted simulations, we can gain valuable insights into the algorithm's performance under various challenging conditions. This will help ensure its reliability and effectiveness in real-world applications. The insights from these simulations will be detailed in the next deliverable, D7.4.

1 Introduction

In the context of monitoring critical infrastructure through satellites and UAVs, the data processing involves extracting valuable insights, detecting potential threats, and assessing the overall condition of the infrastructure. This processed information is then used to make informed decisions regarding maintenance, security measures, and response strategies to mitigate risks and safeguard the critical assets.

High-resolution images and videos of critical infrastructure are captured by satellite and UAV systems. Subsequently, raw data undergoes processing using AI tools to enhance imagery quality and extract pertinent information. This structured approach to data flow ensures that critical infrastructure operators have the necessary tools and insights to respond effectively to challenges and events within their operational environment. The enhanced data is integrated into a user-friendly dashboard interface, empowering operators to visualize and analyze the information effectively. By offering operators a comprehensive view of the infrastructure, this process enables them to make well-informed decisions in response to various issues or events.

1.1 Purpose of the document

This phase aims to continue building a robust foundation for further development, incorporating additional features and models to enhance usability and address various inspection challenges. So far, the technical work has progressed by testing on various public data (available in D9.3), however the next phase will be done in real environment, at CI premises. To this end, this version of the RII tool has been interested in the technical breakthroughs on one hand, while assuring robustness and integration on the other. Furthermore, the technical modules (UAV and Satellite inspections committed a subsequent round of interviews to fully understand the needs and interests of CIs). The physical tests will be conducted in M22 of the project and reported in D7.4.

The progression towards subsequent iterations (D7.4-D7.6) signals a commitment to continuous improvement and innovation in remote inspection tools for critical infrastructures within the SUNRISE project's framework.

1.2 Relation to other project work

In the first phase of development, documented in SUNRISE WP7's D7.1[41] and D7.2[42], comprehensive tools and methodologies were introduced for remote inspection of critical infrastructures. The modular design of satellite and UAV footage inspection tools allowed for independent refinement of components, ensuring flexibility and adaptability. Laboratory validation confirmed successful implementation, showcasing advancements in computer vision and text analysis fields for real-world solutions. The integration of UAV platform hardware and software components laid a foundation for efficient inspections, while user-centric design principles were applied to develop intuitive interfaces.

Work is currently underway to apply some of the UAV-based inspection system solutions in the context of WP6. WP6, which is responsible for developing cybersecurity solutions, has expressed interest in using the computer vision tools deployed by WP7 to correlate physical events with cybersecurity-related events. The open-vocabulary and zero-shot features of the detection models and Visual Large Language Models (V-LLMs) implemented in the WP7 tool modules provide valuable information for these tasks as well. Proof of concept tests have been conducted using videos provided by WP6, yielding promising results.

1.3 The changes from D7.2[42] to D7.3

Table 1: Differences between D7.2[42] and D7.3 summary.

Section in D7.3	Section in D7.2[42]	Differences
Executive Summary	Executive Summary	Updated
1 Introduction	1 Introduction	Text in subsection
1.1 Purpose of the document	1.1 Purpose of the document	Updated
1.2 Relation to other project work	1.2 Relation to other project work	Updated
1.3 The changes from D7.2[42] to D7.3		**New section**
1.4 Structure of the document	1.3 Structure of the document	Updated
1.5 The description of end-users using the RII tool		**New section**
2 Satellite inspection tool	2 Satellite inspection tool	Updated
2.1 General context	2.1 General context	Updated
2.2 Architecture: high level design	2.2 Architecture: high level design	Unchanged
2.3 Tool modules description and lab validation	2.3 Tool modules description and lab validation	Text in subsection
2.3.1 Infrastructure change detection	2.3.1 Infrastructure change detection	Updated
2.3.1.1 Change detection data	2.3.1.1 Change detection data	Updated
2.3.1.2 Methods overview	2.3.1.2 Methods overview	Updated
	2.3.1.3 Methods for change detection	Merged in 2.3.1.2 Methods overview with updates
2.3.1.3 Experiments		**New section** (partially based on previous 2.4.1.2)
2.3.1.4 Future work		**New section**
2.3.2 Vegetation monitoring	2.3.2 Vegetation monitoring	*Text in subsection*
	2.3.2.1 Satellite Providers	Merged in new 2.3.2.1 Vegetation Height Data
2.3.2.1 Vegetation Height Data	2.3.2.2 Vegetation Height Data	Updated with content from previous 2.3.2.1 Satellite Providers
2.3.2.2 Methods overview		New content
2.3.2.3 Experiments	2.3.2.3 Data collection	New content (partially based on previous 2.3.2.4)
2.3.2.4 Future work	2.3.2.4 Data processing	New content
2.4 Applicability for the pilots		New content
2.5 Deployment	2.5 Deployment	Minor change
3 UAV inspection tool	3 UAV inspection tool	Updated
3.1 General context	3.1 General context	Minor changes

Section in D7.3	Section in D7.2[42]	Differences
3.2 Architecture: high level design & use cases definition	3.2 Architecture: high level design	Updated
3.3 Tool modules description	3.3 Tool modules description	Minor changes
3.3.1 Object Detection and Semantic Segmentation	3.3.1 Object Detection and Semantic Segmentation	Major changes
3.3.2 VQA	3.3.2 VQA	Updated
3.3.3 3D Virtualization	3.3.3 3D Virtualization	Updated
3.3.4 Anonymization		New content
3.4 Tool modules lab validation	3.4 Tool modules lab validation	Updated
3.4 Tool modules lab validation	3.4 Tool modules lab validation	3.4 Tool modules lab validation
3.4.1 Object Detection and Semantic Segmentation	3.4.1 Object Detection and Semantic Segmentation	**Major changes**
3.4.2 VQA	3.4.2 VQA	Updated
3.4.3 3D Virtualization	3.4.3 3D Virtualization	Updated
3.4.4 Anonymization		New content
3.5 Deployment	3.5 Deployment	**Major changes**
3.6 UAV platform lab integration	3.6 UAV platform lab integration	
3.6.1 Aerial Vehicle Hardware Specifications	3.6.1 Hardware Specifications	Major changes (UAV has been changed but rest H/W is the same)
3.6.2 Internal Units' Connections and Communications	3.6.3 Internal Units' Connections and communications	Unchanged
3.6.3 Relay Drone System	3.6.4 Relay Drone System	Major changes
4 User interface for remote infrastructure inspection	4 User interface for remote infrastructure inspection	Unchanged
4.1 General context	4.1 General context	Updated
4.2 Architecture: high level design	4.2 Architecture: high level design	Unchanged
4.2.1 Internal Components - Backend Coordinator	4.2.1 Internal Components	Updated
4.2.2 Database security	4.2.2 Technical Specifications	New content (partially based on previous 4.2.1)
4.2.3 Rest API protocol of Dashboard UI		New content (partially based on previous 4.2.1)
4.2.4 Two-Factor Authenticator of Dashboard UI		New content
4.2.5 WebSocket of Dashboard UI		New content
4.3 MQTT Integration	4.3 Dashboards mockups	New content (partially based on previous 4.4)

Section in D7.3	Section in D7.2[42]	Differences
4.3.1 Integration with satellite component		New content (partially based on previous 4.4)
4.3.2 Integration with UAV component		New content (partially based on previous 4.4)
4.3.3 Legacy systems integration		New content (partially based on previous 4.4)
4.3.4 Legacy systems integration security		**New section**
4.4 Deployment	4.5 Deployment	Updated
4.4.1 Security on MQTT protocol		**New section**
5 Pilot trials execution	5 Pilot trials execution	Updated
5.1 Elektro-Slovenija, d.o.o. (ELES)		**New section**
5.2 Ministry of infrastructure of Slovenia (MZI)		**New section**
5.3 Hydro Dolomiti Energia, s.r.l. (HDE)		**New section**
5.4 Designated areas		**New section**
5.4.1 Piloting areas in Slovenia		**New section**
5.4.2 Piloting area in Italy		**New section**
5.4.3 Piloting area in Spain		**New section**
6 Conclusions		Updated
References		Updated
Annex I: Satellite inspection API User Guide		New content
Annex II: UAV Image Processing API System User Guide		New content

1.4 Structure of the document

This document is divided into six (6) primary chapters, including the current one, reflecting the purpose of the document, its framework within the project, and the structure of its contents. The other five include:

- Chapter 2 provides the updates of the high-level design of the satellite inspection tool's architecture for Infrastructure change detection and Vegetation monitoring since the previous deliverable.
- Chapter 3 specifies the work done for the UAV platform integration.
- Chapter 4 discusses the user interface architecture for remote infrastructure inspection and goes deeper to the process details of integration and validation for satellite component, UAV component and CI legacy systems.
- Chapter 5 covers the preparation for execution of Pilot 1 trials on CI areas, updating the content introduced in D7.2[42].

- Chapter 6, final chapter, offers an overall summation of the findings and outcomes of the work detailed in the previous chapters. It should be noted that even though each of the sub-modules has its own distinct chapter, satellite inspection, UAV inspection, and GUI, they are all part of the same remote inspection module, so the integration of the tools at the end of the project must be complete.

1.5 Personas used in the document

Target audience for this document includes end-users for tools developed in WP7. In the context of WP7 definition of end-users refers to RII system from organisations of CI operators. More details about user roles and profiles are given in the chapter 5.5.

2 Satellite inspection tool

This module provides AI-methods for inspection of physical infrastructure using satellite-imagery. We summarize the information given in D7.2[42], to ensure completeness of this document.

The satellite inspection tool provides continuous, easy, and cost-effective way of monitoring and can be used alongside UAV-based inspection (Section 3), which offers a more targeted and detailed inspection. In D7.2[42] we presented two submodules addressing the pilot problems, namely, i) infrastructure change detection and ii) vegetation monitoring. We described the models we developed, the data we used for evaluation and reported our findings.

Here we build upon D7.2[42] by presenting further results and improvements we have since achieved to the two sub-modules (Sections 2.3.1 and 2.3.2) and by discussing their use to the specific use cases in relation to the pilots (Section 2.4).

2.1 General context

Satellite inspection tool provides large-scale and continuous monitoring of CI by automatically processing satellite imagery using computer vision to detect events and changes in the surroundings of CI. Problems requiring remote inspection were discussed with the pilots and were presented in previous deliverables D7.1[41] and D7.2[42]. They can be divided into i) infrastructure change monitoring and ii) vegetation management.

Overgrown vegetation can pose threat to CI and optimisation of tree pruning can provide better reliability of the infrastructure and reduce overall cost. We provide a solution for estimating vegetation height from multispectral satellite imagery (Section 2.3.2).

As identified in previous deliverables, some of the other potential threats to the CI are landslides, leaks, and illegal buildups. We organised several meetings with CIs, to further establish their needs and thus maximise chance of full adoption after the project's end. We provide our findings in section 2.4. We provide a general solution detecting any visual changes in the vicinity of CI infrastructure (Section 2.3.1).

2.2 Architecture: high level design

As introduced in D7.2[42], the architecture of our modules as depicted in Figure 1 consists of the main satellite inspection module and two sub-modules. The main module handles data collection and pre/post-processing. This includes collecting satellite data from various providers based on the provided location and converting the images into a form suitable for the inspection submodules. The two sub-modules denoted as Change and Vegetation and responsible for Change Detection and Vegetation height. They were introduced in Section 2.1 and have not changed from previous deliverables D7.1[41] and D7.2[42].

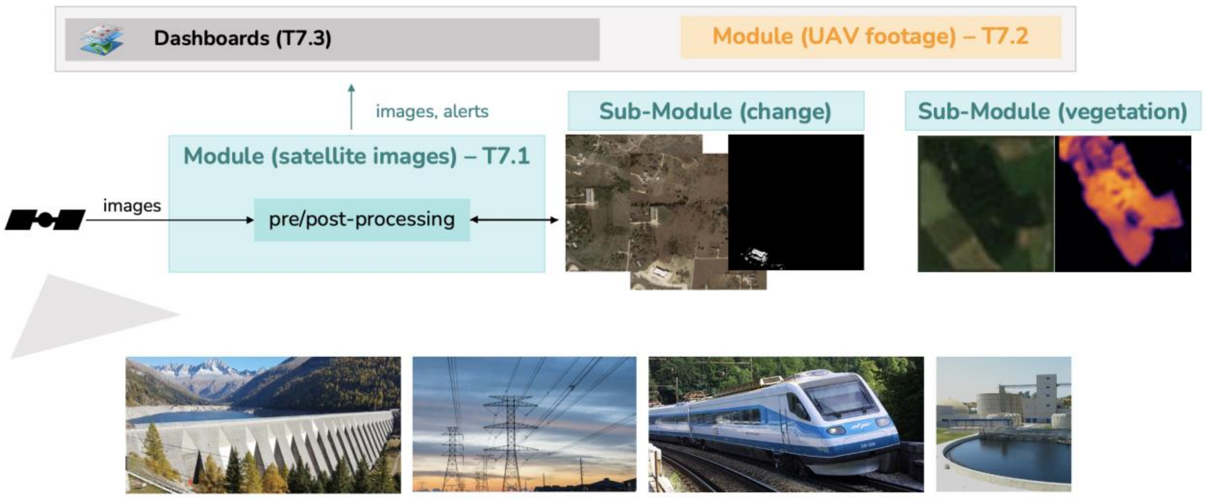


Figure 1: High-level architecture of the satellite-based inspection module and its sub-modules (T7.1).

Remote satellite inspection module receives an API request with defined target area to one of the two sub-modules – Change Detection (CD) or Vegetation Height (VH). We currently support area definition by a point or a polygon. Satellite images are then downloaded from the satellite provider. They are pre-processed to find images with least cloud coverage and to change the data into appropriate shape for the model input. A pretrained model is used to make predictions, which are then post-processed into appropriate output. Vegetation height module returns a map of heights while change detection module returns a binary map with detected changes. See the flowchart of the process in Figure 2.

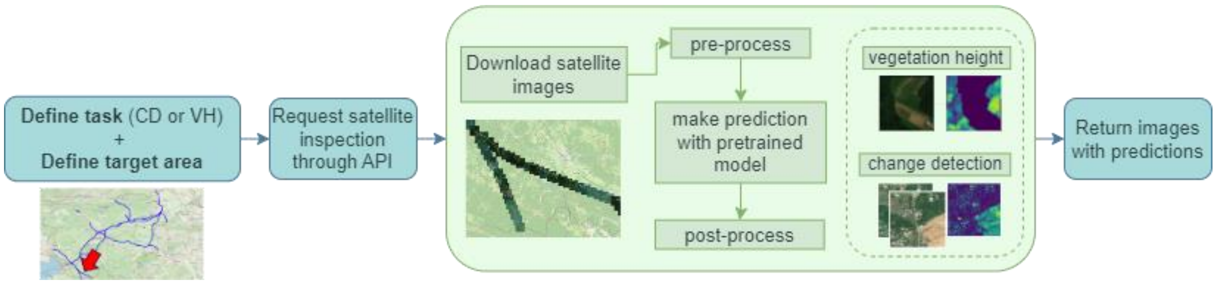

Define task (CD or VH)
+
Define target area
Request satellite inspection through API
Download satellite images
pre-process
make prediction with pretrained model
post-process
vegetation height
change detection
Return images with predictions

Figure 2: Satellite inspection tool request flowchart.

2.3 Tool modules description and lab validation

When developing and testing the methods, we used publicly available datasets, all referenced in their respective sections. While writing this deliverable, we received additional dataset from the CI, concerning change detection, which will be reported upon in the upcoming deliverable, in line with the results of the piloting execution.

2.3.1 Infrastructure change detection

Alongside identification of damage to the critical infrastructure, inspection includes discovering any larger changes in the environment that can pose threats to the systems. Such risks include landslides, vegetation overgrowth, illegal build-ups, etc. These larger events can be detected with high resolution satellite imagery and are the main focus of the first submodule we developed – change detection. The module aims to find any changes to the target areas by analysing images taken at different times. Our main challenge is specifying the type of changes we want to recognise and to build a model that can identify which changes are undesirable.

In D7.2[42], we identified some machine learning models for change detections and datasets, which can be used for training and evaluation. We additionally reported our findings. We mostly focused on evaluation of supervised methods but concluded that unsupervised models would be more suitable for our task. We have since focused on such methods and here we present two unsupervised methods we have developed.

2.3.1.1 Change detection data

Training machine learning models for satellite imagery change detection requires a large dataset consisting of satellite images taken at different times. Additionally, we need annotations of changes, which are used during training (for supervised methods) and evaluation (for supervised and unsupervised methods). There are multiple open-source datasets available. In D7.2[42]. we identified three that were suitable for our case, namely DynamicEarthNet [6], xBD [7] and LEVIR-CD [8].

We continued our evaluation using the LEVIR-CD dataset as it is most used in literature on change detection, and it is therefore easier to compare such models. **LEVIR-CD** dataset (Figure 3) consists of 637 pairs of 1024 x 1024-pixel patches captured by Google Earth. Additionally, it includes a ground truth image denoting temporal changes for each pair. The dataset focuses on building growth. Each pixel in labelled images is marked by binary labels where 1 denotes change and 0 no change in building status.

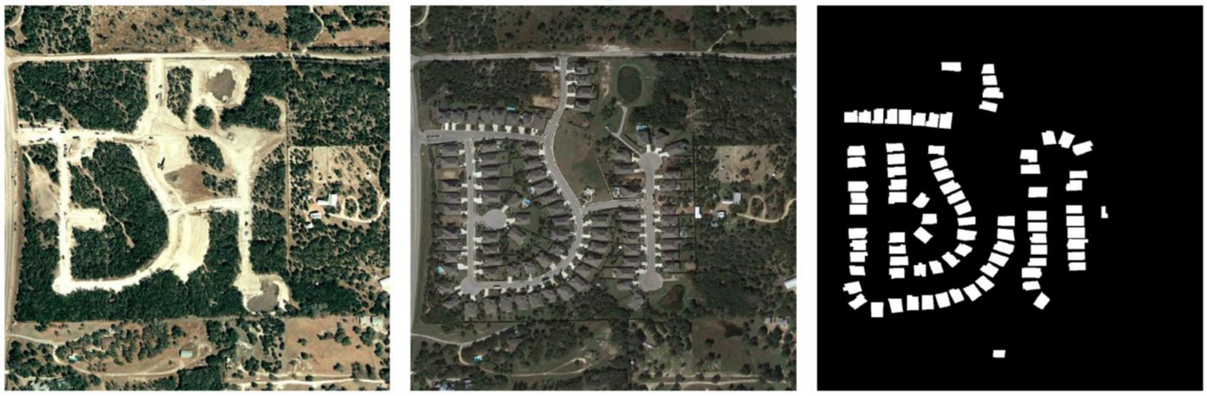


Figure 3: Illustration of LEVIR-CD dataset [8]. Left and middle show images taken at two different times. Right image is labelled with building changes.

Openly available datasets are mainly used to provide a benchmark for comparing different methods for change detection and to compare results to those achieved in literature. One of our use cases focuses on detection of illegal buildups, for which datasets like LEVIR-CD provide a good evaluation. However, since we additionally want to detect other types of changes to the environment, such datasets are not well suitable for training and evaluation. To evaluate the methods specifically for our use cases we require change detection data from our target areas with the type of changes we want to detect. Section 2.3.1.4 describes how we plan to obtain such data.

2.3.1.2 Methods overview

There are two main approaches to the satellite image change detection task: 1) supervised, where machine learning models learn from labelled data with annotated temporal changes and 2) unsupervised, where such labels are not needed during training phase. In D7.2[42] we focused on supervised methods. We evaluated two methods which take two images as inputs (ChangerEx [9] and BiT [10]) and two methods taking three or more images (3-D U-Net [11] and U-TAE [12]). In bi-temporal setting ChangerEx outperformed BiT on the LAVIR-CD dataset. We tested the models taking multiple images as inputs on the DynamicEarthNet dataset.

Supervised methods require extensive labelled data, which are difficult to obtain, since it requires human annotators to label each pixel. In previous section we identified some annotated datasets, however, we do not have any such data available for our specific use case. Additionally, we are looking for novel changes in the environment, while annotated data normally focuses on specific changes such as new buildings. Such problems are better addressed by unsupervised models. In D7.2[42] we presented one such model, namely Change Detection based on image Loss Reconstruction (CDLR) method [13].

CDLR model works by reconstructing the original image from its generated augmentation. During training it therefore only uses needs single-temporal images, as it learns on artificially generated augmentations. During inference, the model detects changes between two input images by finding areas with high reconstruction loss.

We have developed another method for unsupervised change detection. It consists of two components:

1) Image encoder: Extracts features useful for predicting change detection.
2) Cosine similarity: Given two image encodings, compares them by cosine similarity, a measure of similarity between two vectors. We find a threshold for the similarity which determines whether to label a pixel as a change.

Since our task of remote sensing change detection is closely related to image segmentation, we utilized such methods as our starting point. The objective of image segmentation techniques is to partition the input image into multiple segments to detect separate objects contained in the image. Such machine learning models first encode the image through a deep neural network to extract useful features.

Our approach for feature extraction is based on latent diffusion models (LDM) [14], which were originally designed to synthesise images from text. Diffusion models work by de-noising image given noisy image and text input. Since text is included in the training process they produce good semantic internal representations of images. Such models have therefore been repurposed for other tasks in computer vision. One such example is ODISE model [15] for image segmentation. We follow the approach from [15] for the first component in our method - image feature extraction. Given an input image, the model first generates an implicit text embedding. Text embedding and original image with added noise are processed through a pretrained LDM to produce the feature extraction image.

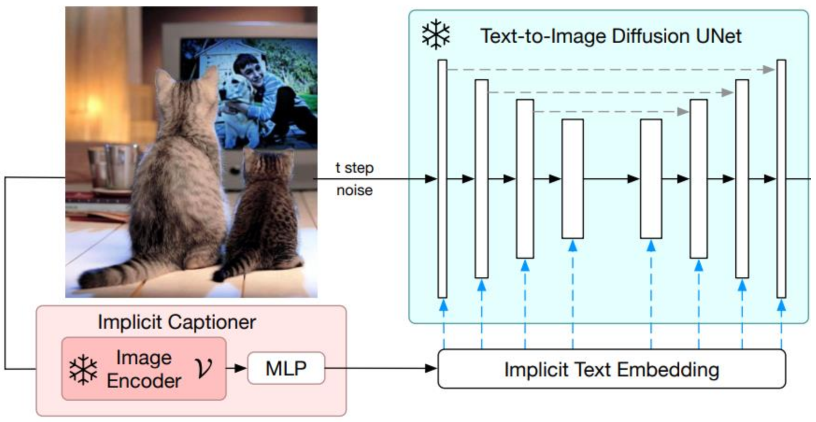


Figure 4: Image feature extractor of the ODISE model [15].

2.3.1.3 Experiments

Evaluation metrics

To compare the machine-learning models we utilize the metrics of accuracy, precision, recall and F1. Accuracy measures the overall correctness of the algorithm while precision shows what percentage of predicted changes were in fact correct. Recall (or sensitivity) on the other hand measures the ability to report all changes, i.e. what percentage of actual changes were detected by the model. Since recall and precision are somewhat opposite scores, F1 score balances between them. A high F1 score therefore indicates the model can both detect all the changes, and we can be confident that all the predicted changes are in fact correct.

Another metric commonly used to evaluate performance of object detection or segmentation algorithms is Intersection over Union (IoU), also known as Jaccard index. It compares the predicted change region to the true change region by calculating the ratio between their intersection to the union. IoU value ranges from 0 to 1, where 0 indicates no overlap between the region and 1 shows perfect overlap.

Given the disproportionate abundance of negative labels compared to positive ones at the pixel level, we additionally employ the use of mean F1 score (mF1) and mean intersection over union (mIoU). The mF1 approach separately calculates two F1 scores by considering “changed” or “not-changed” labels as positive labels and computes their mean value. mIoU is calculated in a similar way.

Quantitative results

Two unsupervised methods for change detection CDLR and method based on latent diffusion models with cosine similarity (LDM-CD) were evaluated on test part of LEVIR-CD dataset. We implemented CDLR method based on the implementation from the [13]. For LDM-CD we used the implementation of the backbone from [15]. We then applied cosine similarity to detect changes and used threshold of similarity of 0.2 to mark a given pixel as changed.

Results are shown in Table 2.

Table 2: Quantitative results of the CDLR and LDM-CD methods on LEVIR dataset.

Method	Accuracy (%)	mF1 (%)	mIoU (%)
CDLR	83.6	49.4	43.8
LDM-CD	61.1	53.6	50.1

Comparing CDLR and LDM-CD methods, CDLR produces results of bigger accuracy, meaning it makes fewer mistakes overall. Accuracy by itself is not a great measure since it doesn’t consider the balance between the positive and negative labels. In change detection algorithms, the number of pixels with no change is in general much higher that pixels where change occurs. It therefore makes sense to consider other metrics as well. mF1 and mIoU are more meaningful metrics. LDM-CD has higher mF1 and mIoU scores, meaning it is better at handling this class imbalance.

Note that these results are not quite as good as those achieved by supervised methods – ChangerEx gave F1 score of 91.77% on LEVIR dataset as reported in D7.2[42]. However, this is expected. LEVIR dataset only focuses on building changes, which can be learned well by supervised methods, while unsupervised can detect other types of changes. Additionally, the bitemporal images in LEVIR dataset show significant land-use changes with the time difference between the images anywhere from 5 to 14 years. On the other hand, our method intends to detect any changes happening in the span of a few weeks or months. We nevertheless evaluated on the LEVIR dataset, since it is widely used in literature, and we can compare our novel method LDM-CD to the baseline CDLR. We plan to evaluate the methods on other openly available datasets, which provide different types of changes. To get a better picture of usefulness of the methods for the CI operators, we will mostly focus on evaluation at the target areas as described in the next section.

2.3.1.4 Future work

The limitation of our current work is the question of applicability of these methods to our specific use case. For that we must perform evaluation on the target data the model will be used on. We obtained limited data on historical detection of illegal buildups from pilot partners. Additionally, we hope to obtain historical data of landslides in Slovenia from e-plaz[43]. We need to evaluate, whether such changed can be observed by the satellite images given the limited resolution and if our models can detect such changes. Provided such evaluation is successful, we will proceed by producing a dataset with artificially in-painted changes, which will be obtained with generative AI methods. This will allow us to produce a more meaningful evaluation and to potentially fine-tune our model on target data.

2.3.2 Vegetation monitoring

The second sub-module we developed is vegetation monitoring where we are trying to identify and predict vegetation overgrowth around critical infrastructure. In D7.2[42] we presented the data used to train the models, the methods we used and their evaluation. We have since tested additional methods and were able to improve our results. Here we describe the methods for achieving the best results.

2.3.2.1 Vegetation Height Data

Our modules use satellite images for making predictions. In D7.2[42] we identified multiple satellite image constellations and evaluated them according to their cost, access to historical archives and spatial resolution. We noted ESA Sentinel-2 and Planet PlanetScope as the two potential providers. We have since focused on Sentinel-2 since it is readily available.

The output of our method is pixel-wise vegetation height prediction. Training the model requires abundant ground truth data. In D7.2[42] we identified Vegetation Height Model (VHM) by National Forest Inventory [16], which stores vegetation height for entire Switzerland and was calculated using digital aerial images. We matched the images with the Sentinel-2 satellite imagery and created training and test datasets.

Our target areas have similar vegetation characteristic to the Swiss training data. We must, however, additionally evaluate the model specifically for our use case, which we hope to do by obtaining vegetation height data of the target areas.

2.3.2.2 Methods overview

The machine learning methods for our task make pixel-wise vegetation height predictions. Making decisions on pixel level is also a characteristic of image segmentation task, where the models classify each pixel by the type of object it presents. Therefore, the methods we tested for our task are based on such segmentation models.

Similarly to our methods for change detection, such models have two components. The backbone model, which extracts useful features and head, which utilizes these features to make vegetation height predictions. In D7.2[42] presented four architectures – Unet [2], DeepLabv3 [17], Swin Transformer [18] and ConvNeXt [19], of which UNet and its variations seem most promising.

Since D7.2[42] we have evaluated additional backbones, some of the most successful are:

- **EfficientNet B8** [1] is a variation of UNet [2][3] – a convolutional neural network characterized by its U-shaped architecture. It consists of a contracting path which decreases spatial dimensions and captures context and an expanding path to reach precise image segmentation with the same dimensions as input image. There are different architectures that can be used for the two paths. We use EfficientNet B8 [1] as backbone. EfficientNet uses compound scaling to uniformly increase the dimension of the model. Increasing the size of the model we obtain models EfficientNet B1 to B8.
- **Riad** [3] is an anomaly detection method based on UNet architecture. It randomly removes regions from the input image and tries to reconstruct the missing parts.
- **Feature Pyramic Netword (FPN)** [4] is a method for feature extraction used for tasks such as object detection. It composes of two pathways 1) bottom-up steps where we scale the image down and 2) top-down steps where we scale it up. Each stage in the top-down pathway is used to make predictions.
- **Pyramid Scene Parsing Network (PSPNet)** [5] is another method for image semantic segmentation. It produces image feature extraction by concatenating a feature map extracted by a CNN with the context extracted by the pyramid pooling module.

Continuing from the work in D7.2[42], we used Mean Absolute Error (MAE) as the training loss and evaluation metric.

2.3.2.3 Experiments

We tested the additional methods on the test split of the VHM data, following the evaluation method from D7.2[42]. We use Mean Absolute Error (MAE) metric, which measures the average absolute difference between predicted and actual vegetation heights. Results for models described in previous section are displayed in Table 3. We additionally include the MAE result of UNet, which uses ResNet50x4 as the backbone and was the best performing model in D7.2[42]. Note that we were able to improve the MAE by 0.1 meters, with RIAD and EfficientNet B8 performing best. Since EfficientNet is more widely used for varieties of tasks in computer vision, we will in the future focus our attention on this model.

Table 3: MAE results (in meters) for newly tested models

Model	RIAD	EfficientNet B8	FPN	PSP	UNet
MAE [meters]	1.57	1.62	2.02	2.31	1.7

Note that the reported values show the absolute error averaged over all the areas in the test set. While the vegetation profile in Slovenia is similar to that in Switzerland, we need to consider that the profile in the specific target areas might be different – the vegetation in the immediate vicinity of CI is low. We therefore need to narrow our test set to the target areas.

Figure 5 shows the qualitative results of EfficientNet B8 model. It displays the visible specter of satellite image on the left, the ground truth vegetation height from Vegetation Height Model [19] in the middle and model's prediction to the right. It shows a variety of landscapes with urban area at the top row, a combination of urban and agricultural areas in the middle and forested area in the bottom row.

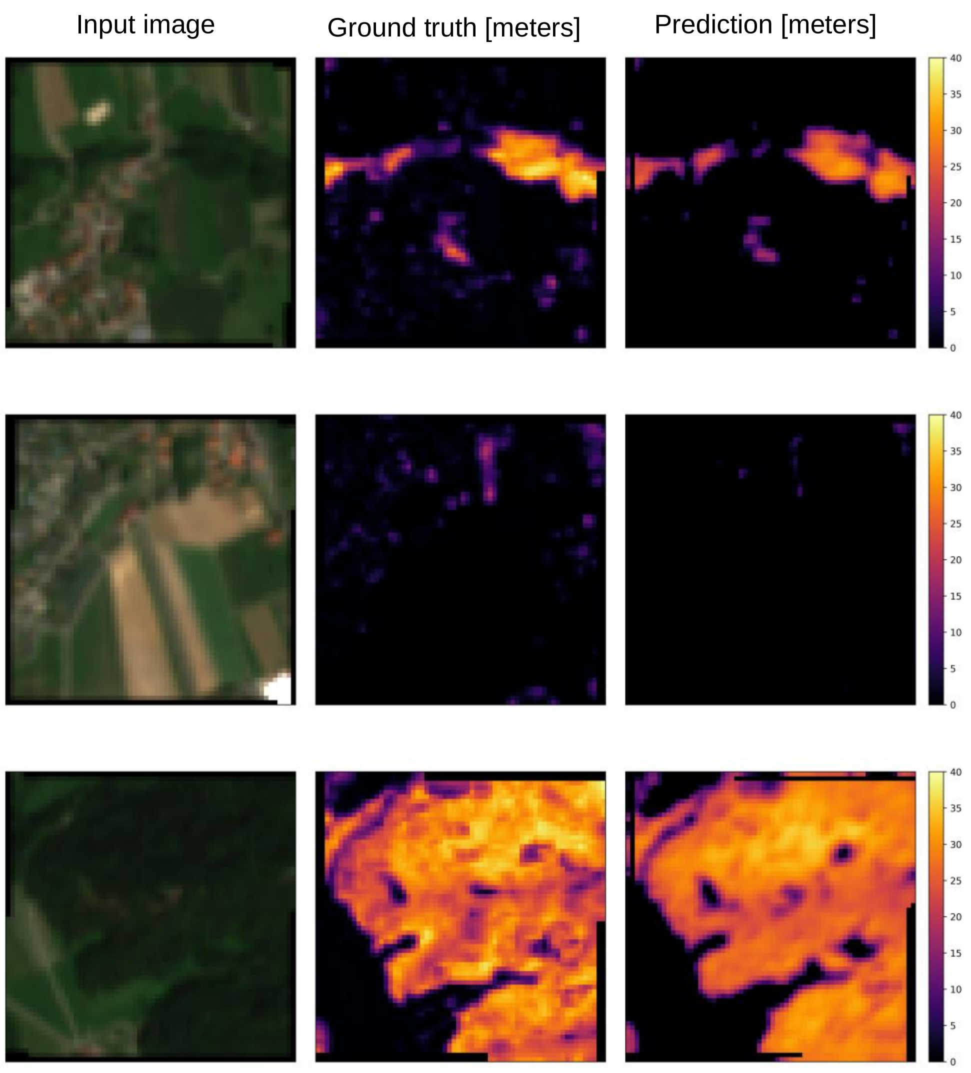


Figure 5: Qualitative results of EfficientNet B8 [1].

The left column shows the visible specter of the Sentinel-2 image, middle column shows ground truth vegetation height from VHM [19] and the right column shows prediction of our model.

2.3.2.4 Future work

We received a small sample of LIDAR data provided by the partner ELES, which will serve as an initial evaluation of our models in the target areas. A more in-depth evaluation will be conducted on LIDAR data obtained from the first pilot trials. Additionally, we will assess whether such data can be used for fine-tuning of our models.

2.4 Applicability for the pilots

Before we develop the toolset further and invest additional resources in research and validation of the models in the field, we as tool developers, must ensure that the developed functionalities bring about the impact and add value to the CIs. To this end, we invested significant resources in research and bilateral meetings with CIs and further drilled down into their needs and wants. This entailed meetings not only with personnel directly involved in the project, but also with more operative personnel, which have a direct connection with the fieldwork. Here we discuss the applicability of our methods for the specific problems of the pilots.

2.4.1 HDE

The execution of additional talks with HDE proved that the viability of using satellite imagery is very limited. Namely, they are interested in detecting clogged grates, presence of excessive sediments in riverbeds near the weirs, problems with their dams, etc. We entertained this idea and pursued the satellite imagery of the highest resolution, obtained from Airbus Pléiades Neo constellaton. However, after the evaluation by HDE, these satellite images, despite their high resolution of 30cm, are still unsatisfactory for such purposes. Furthermore, obtaining such images for commercial use would also result in a high cost of the method (they were provided to us free of charge, however their cost is in the range of 50.000 EUR). Deploying a solution with such a high cost would result in a recurring financial burden for HDE, that they are unwilling to accept, even if the imagery was of satisfactory resolution.

2.4.2 SZ

The vegetation height model developed on the publicly available data [16] works well and has been validated using the dataset itself as ground truth. We, however, delved further in the requirements of the SZ, before we go into full-scale validation, which also requires additional investment from their side. Legal framework of Republic of Slovenia defines 100-meter-wide area from the tracks as an area where building or trees are not permitted (Zakon o varnosti v železniškem prometu - ZVZelP-1 - Article 26). All the special cases (the tracks are e.g. in the narrow valley) are individually evaluated and risk analysis is carried out. Looking at this from this perspective, SZ finds it borderline useful to implement fully-fledged model now. However, the model and the work done is accepted and ready to be used if circumstances change.

Additionally, they are interested in the change detection module – specifically detection of imminent landslides. Together, we reached the conclusion that detection of landslides using just satellite imagery is not useful. They are, quite simply, detected faster using traditional tools, observers, even trains because satellites have quite long spatial resolution. However, using satellite imagery to detect changes and potential landslides, where some changes have occurred and may predict a landslide happening soon, is very useful to SZ. We will pursue a solution in this direction.

2.4.3 ELES

Quite similarly to SZ, we evaluated the usefulness of the developed vegetation height model with ELS. They have similar legal protection (Energetski zakon (EZ-1); Article 468) which defines 40 metres on each side of the power lines as the band where there should be no buildings or vegetation (trees). Given this area is clean in the sense there are no trees, we discussed a variant where there is fast-growing vegetation that could present a concern. However, the cases of this are practically non-existent, rendering this concern a non-issue. Furthermore, we should account for the changing height of the power lines, which could present a security risk. Given the reasoning and low interest in this pilot, we decided the vegetation height is not a concern for ELS.

On the other hand, they identified a problem with illegal build-ups. In practical terms, landowners decide they will build something in the area protected by law, or just fence the area. Special concern present hunting lodges, which are typically rather high and typically have metal roofs. In this case, the ELS is required to intervene. The ability to detect such build-ups is of extreme usefulness to ELS. Thus, we jointly decided to pursue this direction of modelling. We very recently received data on past events, which has yet to be evaluated. We already know that such data is scarce, however we will connect with aforementioned e-plaz and also plan to generate some testing cases artificially, to prove the viability of the approach.

2.4.4 ACO

Additional talks with ACO revealed that the satellite imagery can detect large spills of water - typically if they are large enough to make a stain on the surface. However, such large spills are detected much sooner using the technology ACO already has and takes advantage of the direct control of water pressure in the water supply pipes. Furthermore, it requires the event to be in remote areas because in urban areas, pipe spills and bursts are immediately detected. Thus, we jointly decided to pursue much cheaper and viable technology - use of UAVs.

2.5 Deployment

The deployment methodology did not change from the D7.2[42], hence we are providing just a short note on it.

The inference parts of the developed solutions are available as-a-service via REST API, enabling easy integration into user interfaces. It is deployed to Microsoft Azure Cloud, which provides reliability and scalability. We use models pretrained offline since the methods do not need to be retrained frequently. In the future we will use MLFlow[44] for versioning of the deployed AI model.

3 UAV inspection tool

This section of the document provides an update on the components of the sub-module dedicated to inspecting critical infrastructures using UAV-captured imagery, continuing the work from task T7.2. Maintaining the structure used in the previous deliverable, this section transitions from overarching themes to specific details. It starts with a summary and context of the task, followed by an outline of the tool's general architecture. The narrative then moves into an in-depth examination of the software methods and algorithms developed, their validation and deployment strategies, and concludes with details on the hardware setup and how it integrates with the UAV platform, ensuring content is refreshed while preserving the established format.

3.1 General context

The UAV image-based visual inspection module, as detailed in the previous deliverable, D7.2[42], harnesses the significant advancements in image analysis technology to conduct focused, non-invasive surveys of critical infrastructures (CI), including their structures and specific components. This innovative module integrates UAV-acquired imagery to modernize traditional visual inspections, employing Artificial Intelligence (AI) to streamline data analysis. Utilizing cutting-edge deep learning and computer vision techniques, the module tackles various challenges highlighted by stakeholders, enhancing the efficiency, periodicity, and accuracy of CI inspections.

Depending on the specific requirements of each inspection task, particular visual sensors are selected to capture the necessary data. The primary aim of this module is to automate preventive maintenance tasks for CI, expedite the response to failures, and develop versatile solutions applicable across diverse CI scenarios. These technological solutions are designed to complement and augment satellite-based inspections, as discussed in Chapter 2 of the previous version of this deliverable, by offering the capability to examine extensive areas with heightened precision and detail, particularly in zones deemed high-risk or of critical importance.

As elaborated in Deliverable D7.1[41], this module aims to provide comprehensive solutions for several key areas, like the inspection of catenary networks and power lines; the detection of generic anomalies such as floods, landslides, and fires; the examination of structural or specific components like cracks, corrosion, or leaks in pipes; and the 3D virtualization of infrastructure. These areas represent the core focus of the module, highlighting its potential to significantly impact the maintenance and safety of critical infrastructures through advanced UAV-based visual inspection techniques.

3.2 Architecture: high level design & use cases definition

In this section, the advancements in the architecture implementation and the general tool use cases are detailed, adhering to the foundational principles highlighted in the previous deliverable, D7.2[42]. Principles such as modularity, self-containability, and scalability have been pivotal in guiding the development of the UAV image-based critical infrastructure inspection tool. As previously outlined, the main tools used to implement solutions following these criteria are Docker and REST API. The updated architecture is defined in the Figure 6, capturing the various elements that make up the system and how they exchange data.

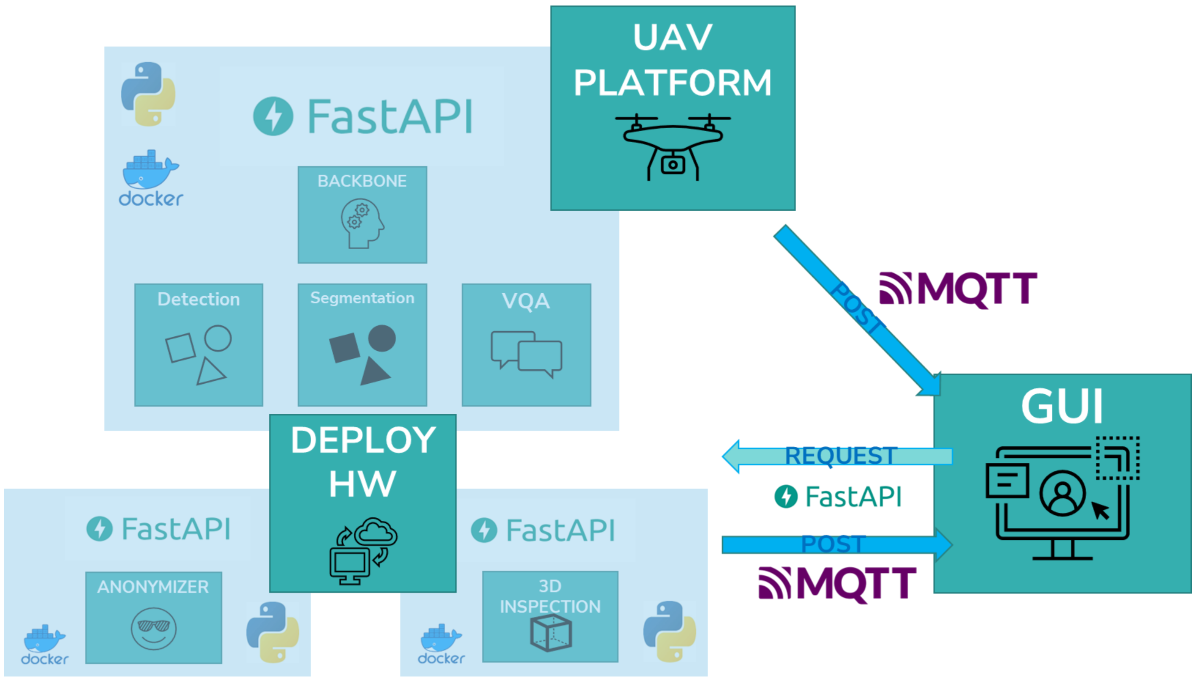
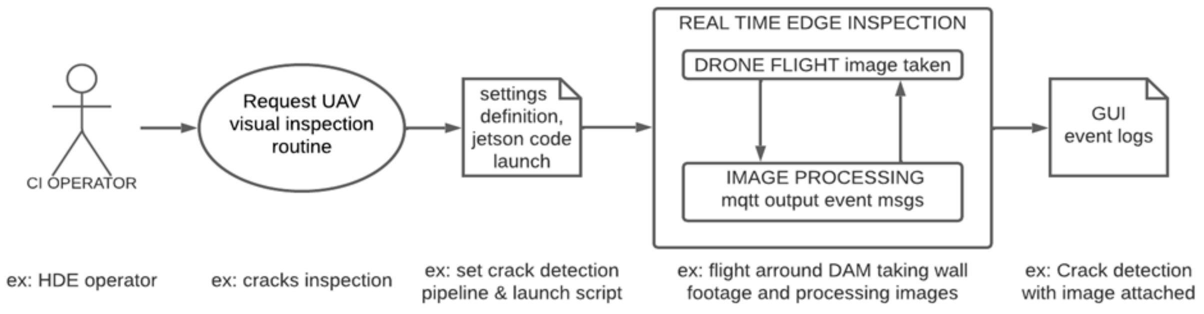


Figure 6: UAV visual inspection tool general architecture.

As illustrated in Figure 6, there are three main entities within the system architecture: the UAV platform, the Graphical User Interface (GUI), and the hardware (Deploy HW) responsible for hosting APIs that are accessible remotely. Communication among these components occurs either through the deployed MQTT broker or by utilizing the request-post protocol of the APIs.

At this stage of the project, the implemented Python code is structured into three distinct APIs to enhance interoperability and allow for their individual use. This modular approach helps prevent library dependency conflicts that could arise within each API. The core functionality resides within the main API, which encompasses the backbone and modules for detection, segmentation, and Visual Question Answering (VQA). The backbone script of this main API orchestrates processes, receives inspection parameters and system inputs, and internally calls other APIs as needed.

The defined architecture allows for the deployment of solutions both to run as a cloud service hosted in the deploy HW, which can be accessed remotely with pre-recorded data, and to operate on an edge device in real time. Following this dual approach, subsequent images illustrate these two defined use cases.



CI OPERATOR
Request UAV visual inspection routine
settings definition, jetson code launch
REAL TIME EDGE INSPECTION
DRONE FLIGHT image taken
IMAGE PROCESSING mqtt output event msgs
GUI event logs
ex: HDE operator
ex: cracks inspection
ex: set crack detection pipeline & launch script
ex: flight arround DAM taking wall footage and processing images
ex: Crack detection with image attached

Figure 7: UAV remote inspection tool live stream processing use case flowchart.

Figure 7 shows the workflow for the use case where the individual responsible for inspecting critical infrastructure facilities wants to perform a real-time inspection routine, looking for events such as cracks or fire. To achieve this, the Critical Infrastructure (CI) operator must submit a request via the main API, specifying the parameters of the desired inspection routine. This involves selecting one of the available presets (simple) or elaborating on which processes and models will be used (more complex). Upon initiating the request, in an ideal scenario, the Unmanned Aerial Vehicle (UAV) would be ready and able to autonomously perform flights along pre-recorded paths, leading to its automatic deployment and the execution of the solution in real time at the edge, with reports or alerts being remotely sent to the GUI. If, however, the UAV is not ready for autonomous flight, it would be necessary to wait until a UAV operator is available to carry out the flight and analyze the images in real time.

The second use case is suitable for all those inspections that do not require a real-time response, that is, where obtaining the inspection report right now or in a few hours/minutes does not pose a problem. Most applications fall within this category. In this case, as in the previous one, the Critical Infrastructure (CI) operator must define the inspection parameters through the API, but this time they will attach pre-recorded images or videos upon which the deployed solutions will be applied, resulting in reports with the detected events. This case is exemplified in Figure 8.

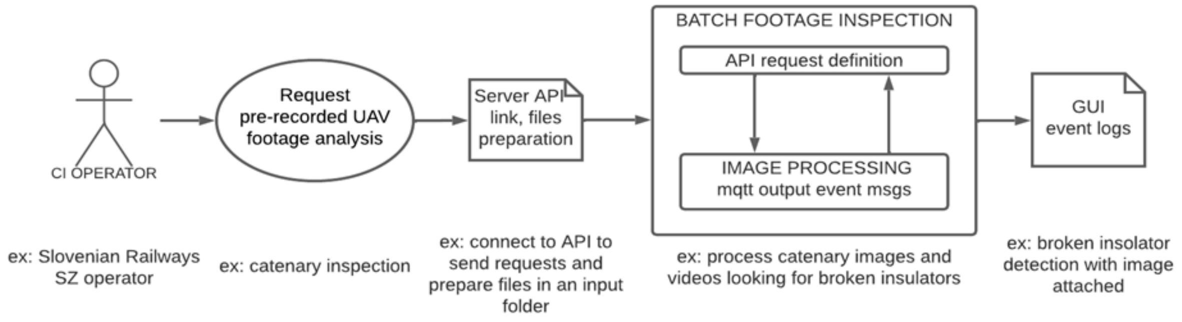

CI OPERATOR
Request pre-recorded UAV footage analysis
Server API link, files preparation
BATCH FOOTAGE INSPECTION
API request definition
IMAGE PROCESSING mqtt output event msgs
GUI event logs
ex: Slovenian Railways SZ operator
ex: catenary inspection
ex: connect to API to send requests and prepare files in an input folder
ex: process catenary images and videos looking for broken insulators
ex: broken insolator detection with image attached

Figure 8: UAV remote inspection tool batch processing use case flowchart.

Thus, the components and processes involved in the current solution for critical infrastructure inspection with UAVs are defined, laying the groundwork for the first round of field tests. Following this testing period, a review of the architecture will be conducted, analyzing the performance in that real-world application context.

3.3 Tool modules description

In this section of the deliverable, there is a revisit of D7.2[42], detailing the enhancements made to the UAV image-based critical infrastructure inspection module. The content has been updated maintaining the format established in the previous iteration, to further refine non-invasive, focused inspections of critical infrastructures. The section unfolds with an in-depth examination of the advancements across specific tool modules such as Object Detection, Semantic Segmentation, Visual Question Answering (VQA), 3D Virtualization, and Anonymization, highlighting the steadfast dedication to augmenting the efficiency and precision of critical infrastructure assessments utilizing state-of-the-art UAV and AI technologies.

The structure of this section has been slightly modified with the incorporation of an additional module, the Anonymization module. As depicted in Figure 6, this new component operates autonomously from the UAV inspection tool's core module. The Anonymization tool is responsible for receiving raw images captured at the facilities of CI stakeholders and anonymizing them to prevent the recording of personal data in the reports generated from the automatic inspection process. This addition is motivated by the Project's commitment to data protection legislation and to thereby avoid GDPR issues.

While this section is primarily intended for theoretical introduction and process description, practical examples of the application of the models introduced in this section can be found in the following section, 3.4, which includes some of the inspection use cases put forward by CI stakeholders.

3.3.1 Object Detection and Semantic Segmentation

The advancement of the detection and segmentation module remains pivotal for the UAV remote inspection tool's progression, providing high-value tools for computer vision applications. As detailed in deliverable D7.2[42], it serves both as a final event detector for incidents like floods or landslides and as a preliminary step in more complex pipelines for object detection, as will be demonstrated in the VQA module in section 3.3.2 of this document.

The dual approach from the initial deliverable version is upheld, leveraging innovative foundational open-vocabulary models like GroundingDINO [23] and SAM-HQ [24] for broad applicability and integrating state-of-the-art computer vision capabilities, while simultaneously employing conventional architectures such as Convolutional Neural Networks (CNN, yolov8) [25] for specific real-time tasks, exemplified by the fire and smoke detection model in D7.2[42].

This update introduces new developments. YOLO-Worldv2 [25], a broad-spectrum model, brings open-vocabulary and zero-shot capabilities with reduced inference times compared to the other implemented options. Furthermore, two specific models for detecting corrosion and cracks in structural elements have been trained, utilizing Roboflow datasets [20], [21], and [22] to enhance model specialization.

The primary metrics used to measure the performance of the models outlined below are mAP0.5 and the F1 score, as they are commonly employed and highly indicative of the model's performance, taking into account True Positives (TP), False Positives (FP), and False Negatives (FN). Equation 1 displays the mathematical formulation of these metrics. The F1 score indicates how effectively the model balances detecting True Positives (TP) while minimizing False Positives (FP).

Equation 1: Precision (A), Recall (B), F1 (C) and IoU (D) formula.

$$Precision = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Positives\ (FP)} \qquad (A)$$

$$Recall = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Negatives\ (FN)} \qquad (B)$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \qquad (C)$$

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \qquad (D)$$

The mAP0.5 refers to the mean Average Precision (AP) for each detected class, considering a detection correct when the prediction and the labeled bounding box (ground truth) overlap at least 50%, i.e., the Intersection over Union (IoU) value is greater than 0.5.

The trained models those aimed at solving a specific problem, are based on a fine-tuning of YOLOv8-seg, in its version with more parameters, the heaviest model, namely model X. This decision is motivated by the fact that YOLOv8 tops the rankings of real-time detection models, is easy to implement and train, and does not require very high hardware specifications. The choice of model X is because processing images in strict real-time is not necessary; being able to process between 1 and 10 frames per second is more than enough to carry out the inspection process satisfactorily.

The first of the two ad-hoc trained models detects and segments cracks in concrete, and for this, a combination of datasets [20] and [21], which are publicly available on the Roboflow platform and include images of scenes representative of the scenarios expected to be encountered in the real pilots, has been used. In the near future, after conducting the project's pilots, it is intended to introduce more data into the set, with the intention of improving the model's performance in our specific use cases by adapting the model's domain. The evolution of the training metrics can be observed in Figure 9, where the graphs of two training processes are attached: in 9A there is a first iteration, with 100 epochs of training, in which signs of overfitting are noticeable, such as the increase in the loss functions or the slight decrease in mAP 0.5. In 9B, the epochs are reduced to 50 to avoid the overfitting issue.

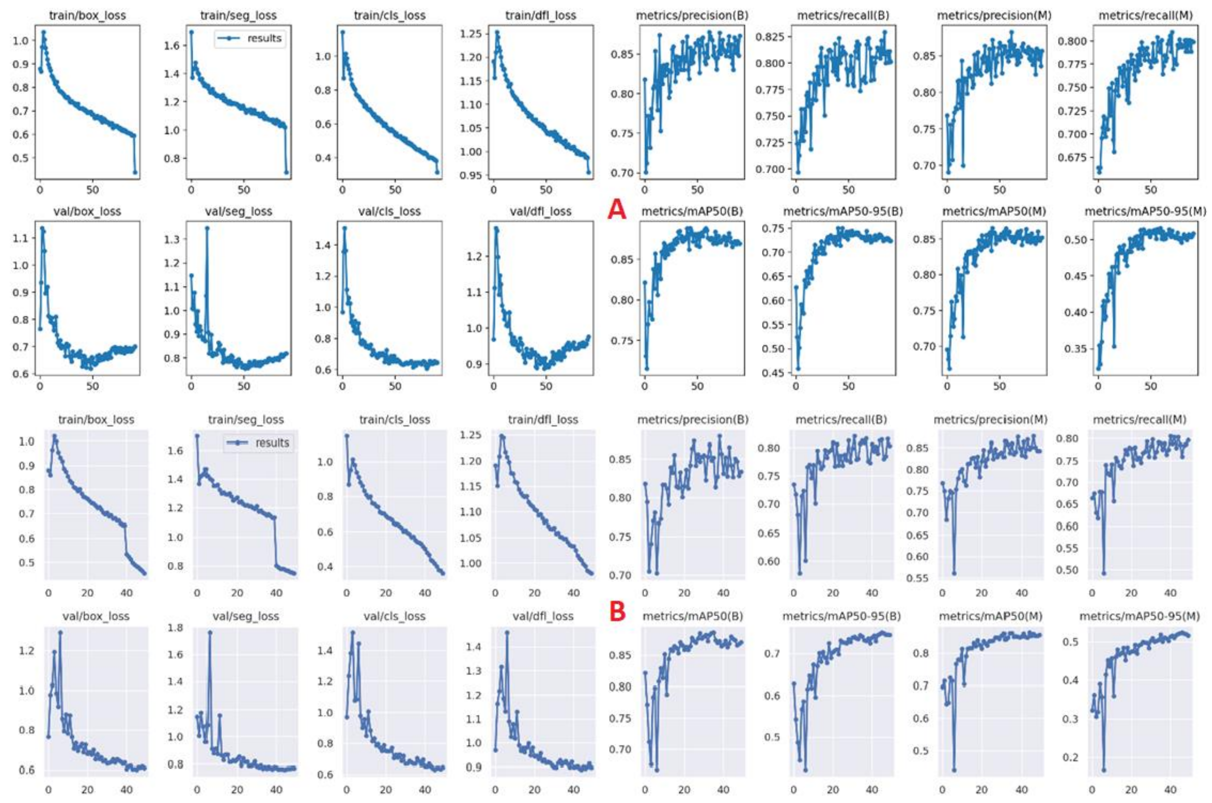


Figure 9: 9A (Upper) first iteration YOLOv8X-seg Crack detection model training metrics in combined datasets [20] and [21]. Signs of overfitting; 9B (Botton) second iteration YOLOv8X-seg Crack detection model training metrics in combined datasets [20] and [21]. No sign of overfitting.

The metrics obtained yield an mAP0.5 of 0.874 and an F1 score of 0.82, indicating a solid performance of the model on the validation set, laying a good foundation for further work on the model in future deliveries and for testing the model with footage collected during the near-future Pilot 1.

The second of the trained YOLOv8x-seg models semantically segments 3 different degrees of corrosion in steel structures: fairly corrosion, poor corrosion, and severe corrosion. The dataset used, [22], is again publicly available on Roboflow and also aims to be as close as possible to the real use case. Similarly to the case of cracks, the intention is to adapt the model's domain with images from our CI stakeholders in future stages of the project, provided that quality labeled images from the pilots can be obtained. The main results obtained are reflected in the confusion matrix and the F1 curve in Figure 10.

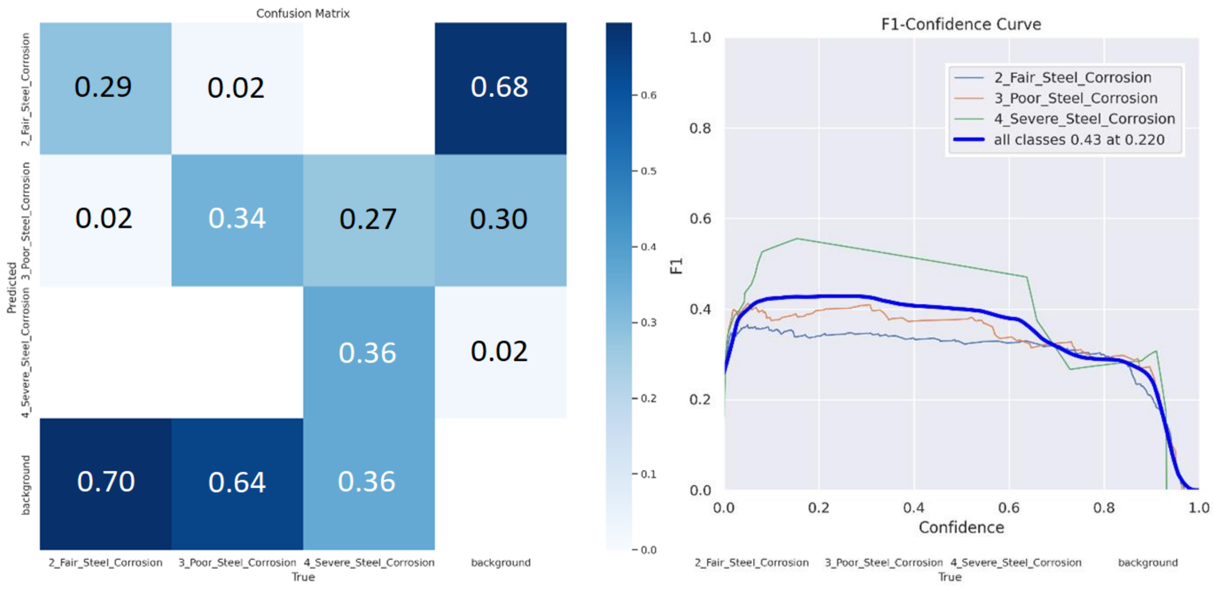


Figure 10: Confusion Matrix (left) and F1 curve (right) resulting from training YOLOv8X-seg in a dataset with three different levels of corrosion annotated, [22].

As can be observed, on this occasion, the results achieved do not reach high levels of mAP or F1, standing at 0.35 and 0.43, respectively. This is primarily due to the complexity of the task and the fact that publicly available datasets are not of the highest possible quality. Despite this, as can be inferred from the confusion matrix, a significant percentage of areas with severe corrosion are detected, comprising 36% that the model labels as this class, and 27% that it labels as poor corrosion. These classes could be used to trigger alerts in our system. Moreover, the severe corrosion class exhibits a very low false positive rate, at only 2%. Therefore, it can be stated that this is a good starting point, considering there is room for improvement in future iterations, with more data in our domain. In section 3.4.1, Figure 17, it is qualitatively evident that the model is capable of correctly detecting a substantial portion of the corroded areas.

Regarding the second approach of this module, which enables the detection of any class by defining any concept as an input to the model through a simple textual prompt, the main update implemented in this period is YOLO-World2. Its key feature is that it facilitates open-vocabulary detection in real-time. Figure 11, taken from the paper introducing YOLO-World, [26], visually captures the process and components involved in this model.

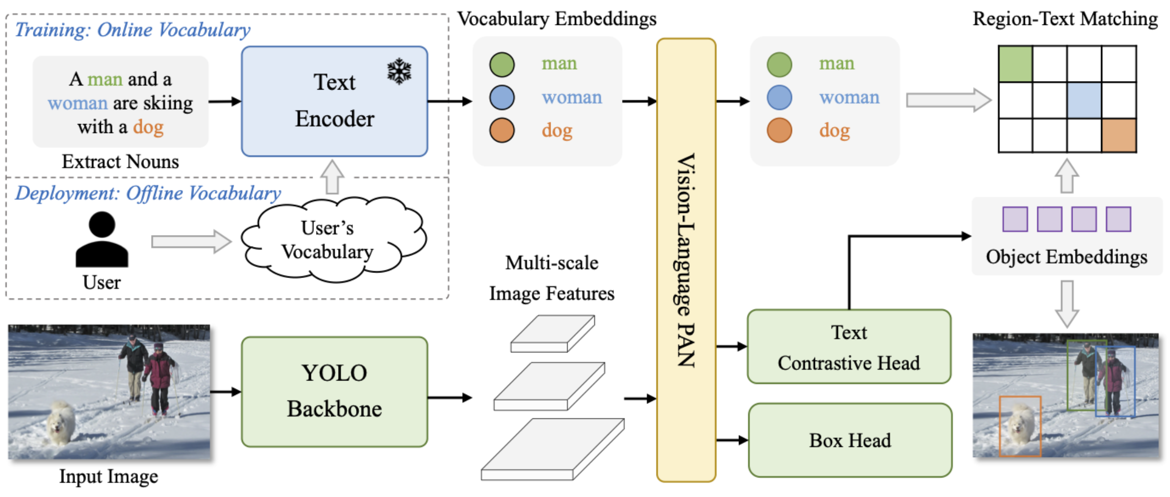


Figure 11: Overall Architecture of YOLO-World. Source: [26].

As the other open-vocabulary and zero-shot models, YOLO-World differs from classic YOLO detectors by using text as input, turning it into a versatile tool that can identify a wide range of objects. As shown in the image, it starts by transforming the given text into a feature format it can work with. Next, it processes the image, breaking it down into features at different scales. It then combines the information from both the text and the image by using Vision-Language model. In the end, YOLO-World pinpoints where objects are in the image and matches them to the terms provided in the text input.

With these new solutions, adding to those already introduced in D7.2[42], the object detection and segmentation module implements and makes available to the user a significant range of options to address their specific inspection problems. The models adapted so far are: Detic [27], X-Decoder [28], GroundingDINO [23], SAM [24], YOLO [25], and YOLO-World [26].

3.3.2 VQA

This second module, aimed at implementing tools based on the disruptive Visual-Language Models (VLM), has evolved the most since the status reported in D7.2[42]. The previous deliverable introduced the BLIP-2 [29] model, which already showed great potential and high applicability. In the update set out this document, two new recently published models are introduced, whose ideas and performance take this VQA module to the next level up.

The first introduced model is not just another newer Multimodal Large Language Model (MLLM) or Vision-Language Model (VLM) designed for Visual Question Answering (VQA), such as BLIP-2, but it also introduces a novel meta-architecture for image search and answering questions about specific objects, which can be highly relevant for inspection tasks. This architecture, introduced in [30] and called V*, showcases several interesting concepts.

One key element in V* is the Language Model-guided object search mechanism, which, as illustrated in Figure 12, mimics the process a person would follow to efficiently find objects by guiding the search, reasoning about the most likely parts of the scene where the searched object might be located. This model can infer, for example, that the most likely place to find a glass in the Figure 12 scene is on the table, and uses the LLM's attention maps when searching for the concept to pinpoint the object's location.

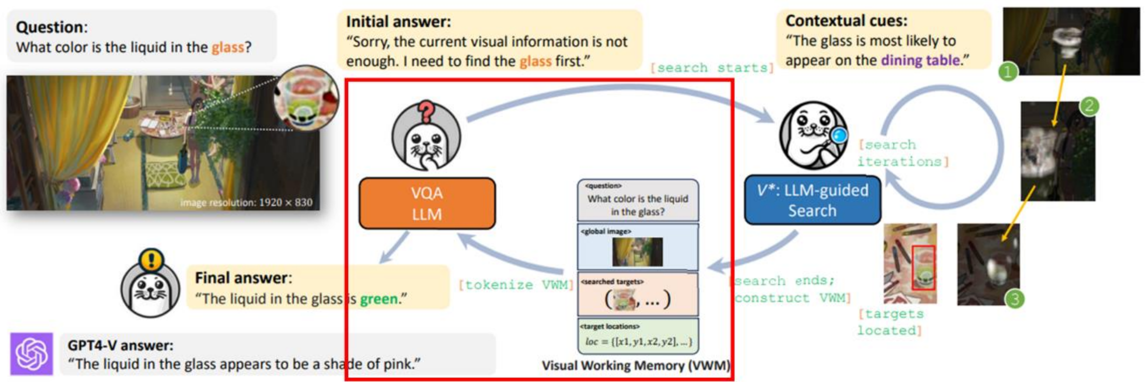

Question:
What color is the liquid in the glass?
image resolution: 1920 × 830
Initial answer:
"Sorry, the current visual information is not enough. I need to find the glass first."
[search starts]
Contextual cues:
"The glass is most likely to appear on the dining table."
VQA LLM
<question> What color is the liquid in the glass?
<global image>
<searched targets>
<target locations> loc = {[x1, y1, x2, y2], ...}
Visual Working Memory (VWM)
V*: LLM-guided Search
[search iterations]
[search ends; construct VWM]
[targets located]
[tokenize VWM]
Final answer:
"The liquid in the glass is green."
GPT4-V answer:
"The liquid in the glass appears to be a shade of pink."

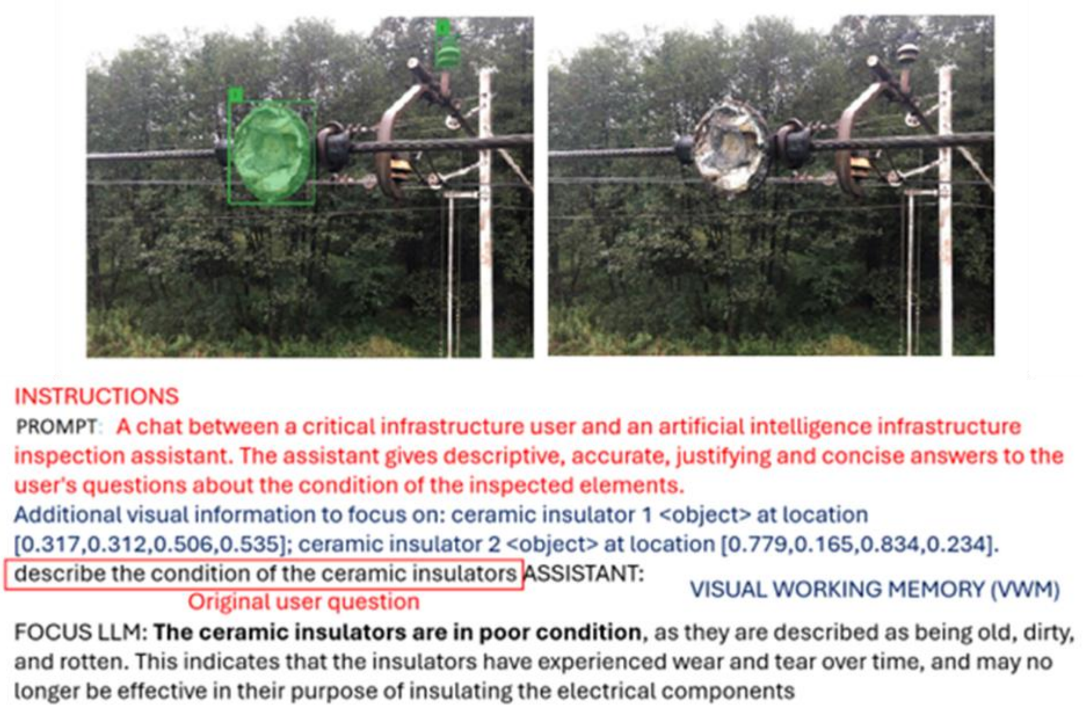
Figure 12: V* overall architecture and concept explanation. Source: [30].

Another significant concept in V* is the Visual Working Memory (VWM), highlighted inside the red square in Figure 12. VWM is a data structure that encompasses four fields and formats the input information for a VQA model fine-tuned to interpret such data structures, maximizing the use of provided information. The VWM fields include the question to be answered, features extracted from the entire image, features from each object of interest's crop, and the position of each object in the image.

Using VWM information to generate input prompts for the VQA model is highly beneficial for answering specific questions about objects of interest, as the VQA model can focus more on the visual features of these objects without losing the context of the entire image. Moreover, this approach addresses one of the main challenges with VLMs: the information loss due to dimensional reduction when representing the entire image with a limited number of extracted features, such as 128. By extracting crops of objects of interest, the proposed meta-architecture not only extracts 128 features from the entire image but can also obtain 128 visual features for each crop of interest, significantly enhancing the information available to the VQA model for answering the question without significantly increasing the required processing.

In the solution implemented for SUNRISE, the part of the LLM-guided search has been omitted since it is designed to find small objects in high-resolution images, and in the case of critical infrastructure inspection, it is more efficient to use models like GroundingDINO or YOLOv8. These models offer better performance and lower computational cost in detecting the objects of interest. For this reason, the output from the previously discussed object detection and segmentation module is used to gather the necessary information to populate the fields of the VWM and feed the proposed VQA LLM in this work, V*.

In order to try to illustrate the developed pipeline in a clearer way, a detailed example of a real use case has been introduced. Figure 13 shows an image from SZ facilities, taken to inspect the ceramic isolators of the catenary on the train tracks. As shown in the upper image, the first step is to detect the ceramic isolators using the detection module. Once the elements to be inspected are detected, the VWM is filled with the user's question, the positions of the objects, and the visual features of the complete image and the object cutouts. Then, with the information from the VWM and the specific instructions of the inspection use case, the input prompt for the V* LLM model, called SEAL VQA, is generated.


INSTRUCTIONS

PROMPT: A chat between a critical infrastructure user and an artificial intelligence infrastructure inspection assistant. The assistant gives descriptive, accurate, justifying and concise answers to the user's questions about the condition of the inspected elements.
Additional visual information to focus on: ceramic insulator 1 <object> at location [0.317,0.312,0.506,0.535]; ceramic insulator 2 <object> at location [0.779,0.165,0.834,0.234].
describe the condition of the ceramic insulators ASSISTANT:

Original user question

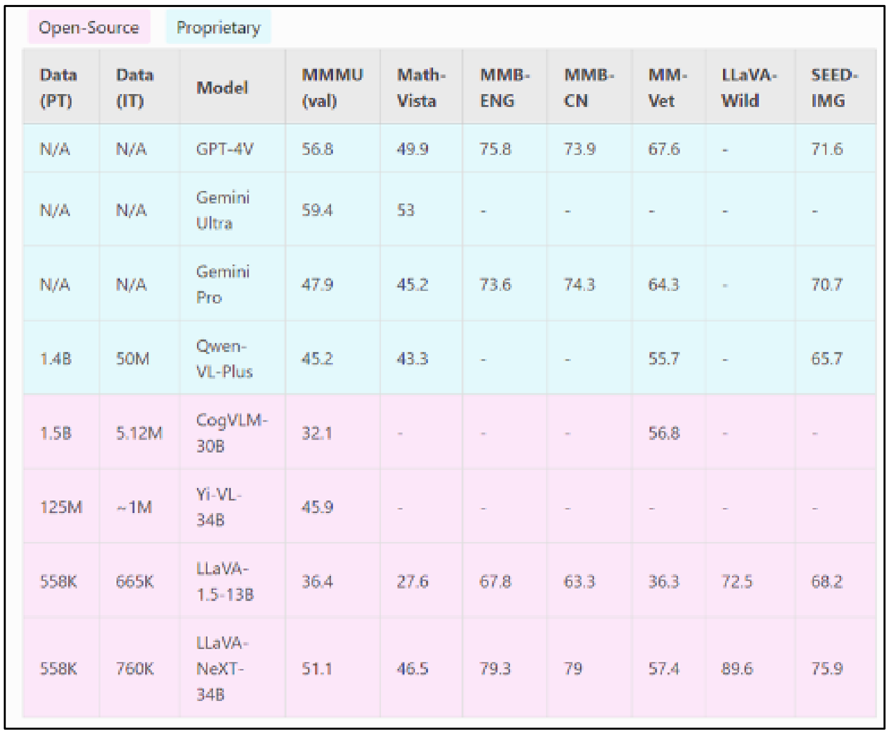
VISUAL WORKING MEMORY (VWM)

FOCUS LLM: **The ceramic insulators are in poor condition**, as they are described as being old, dirty, and rotten. This indicates that the insulators have experienced wear and tear over time, and may no longer be effective in their purpose of insulating the electrical components

Figure 13: Ceramic insulator status inspection in SZ facilities with Focus VQA module. Full prompt used as input for VQA module, including generic context information (red), Visual Working Memory VWM (blue), original user inspection question (red box), and model's answer (black).

The output of the SEAL VQA model, as shown in the text of the Figure 13 under the header "FOCUS LLM," confirms that the isolators are in poor condition, justifying its response with terms such as old, dirty, wear and tear. The image analysis is correct, and together with other qualitative results obtained for other use cases, it can be stated that the solution exhibits a remarkably positive performance.

The second model introduced in the VQA inspection tool is LLaVA-1.6 [31], the recently published update from the LLaVA model family [32]. This model sets a new open-source state of the art in Visual Language Models used for VQA, standing out for its great performance and widespread use. Figure 14 and Figure 15 show the performance comparison of LLaVA-1.6 (LLaVA-NeXT) against the leading proprietary and open-source models, using the current benchmarks for the VQA task. The table has been extracted from [31], and the graph representing the table's data has been created to try to make the information more visual and accessible.


Open-Source Proprietary

Data (PT)	Data (IT)	Model	MMMU (val)	Math-Vista	MMB-ENG	MMB-CN	MM-Vet	LLaVA-Wild	SEED-IMG
N/A	N/A	GPT-4V	56.8	49.9	75.8	73.9	67.6	-	71.6
N/A	N/A	Gemini Ultra	59.4	53	-	-	-	-	-
N/A	N/A	Gemini Pro	47.9	45.2	73.6	74.3	64.3	-	70.7
1.4B	50M	Qwen-VL-Plus	45.2	43.3	-	-	55.7	-	65.7
1.5B	5.12M	CogVLM-30B	32.1	-	-	-	56.8	-	-
125M	~1M	Yi-VL-34B	45.9	-	-	-	-	-	-
558K	665K	LLaVA-1.5-13B	36.4	27.6	67.8	63.3	36.3	72.5	68.2
558K	760K	LLaVA-NeXT-34B	51.1	46.5	79.3	79	57.4	89.6	75.9

Figure 14: Table comparison SOTA in VLM for VQA benchmarks. Source: [31].

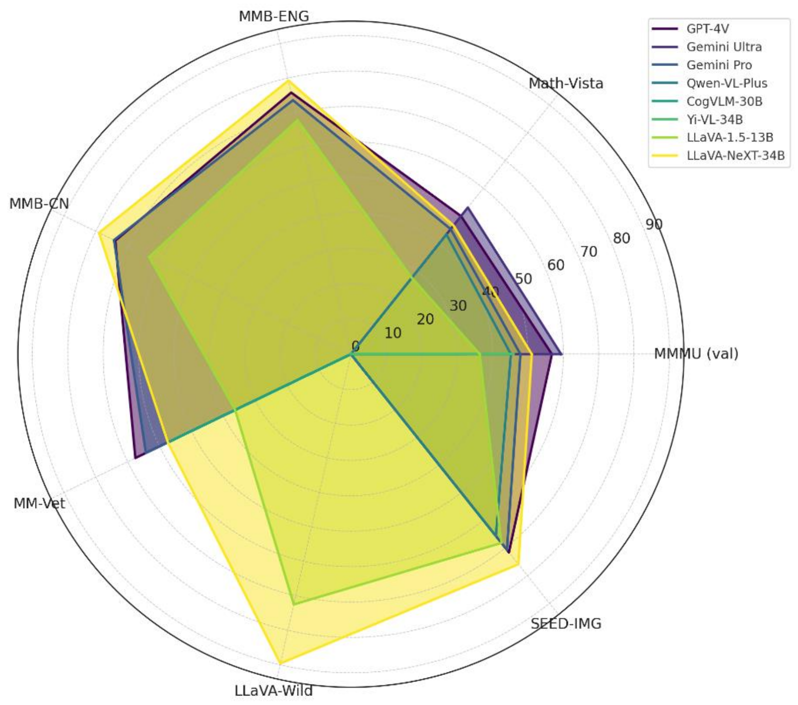


Figure 15: Graph comparison SOTA in VLM for VQA benchmarks, elaborated from [31] data.

As can be deduced from the table and the figure, LLaVA-1.6 is the best open-source model available, and it even outperforms some proprietary models in certain datasets. In addition to its notable performance, it's worth mentioning that the available quantization options make it a model that can be deployed on modest GPUs with as little as 8GB of vRAM.

With these two new solutions integrated into the module, the tool is now equipped to tackle the inspection of a wide range of different issues. Besides the examples included in section 3.4, it will continue to be tested during the first phase of piloting. The models adapted in this module so far are: BLIP-2 [29], V* (SEAL VQA) [30], and LLaVA1-6 [31].

3.3.3 3D Virtualization

There have been no updates in the implementation of this module since the previous deliverable. Therefore, it is recommended to review this section in the aforementioned Deliverable D7.2[42] to revisit the complete pipeline of this module. This pipeline involves the reconstruction of the UAV's path using COLMAP, background extraction through GroundingDINO and SAM-HQ, and the training of a NeRF neural model capable of rendering the scene from any angle and position provided as input.

While no efforts have been made to incorporate new functionalities or models into the tool, it has been employed in real-world scenarios at the facilities of Critical Infrastructure (CI) stakeholders and integrated with the GUI developed in Task T7.3, which is discussed in section 4 of this document.

This process of preliminary testing and integration, before the tests to be conducted with the data collected in the first off-lab pilot at M20 of the project, has identified two areas for improvement. Firstly, the process is semi-automated; the API receives raw videos from the Points of Interest (POI) flights, autonomously calculates the camera's path, and extracts objects/structures of interest from the scenes (background extraction). However, the process of launching the NeRF model's training and verifying that the results are adequate remains manual. This could pose a challenge if a fully automated process is desired that directly outputs the model on the graphical interface. This issue could be addressed by adding an intermediate manual train and validation step. Secondly, another significant area for improvement has been identified, and that is that the files generated as output, in 3Dmesh format, are also very large, which makes them difficult to view in the GUI tools.

In addition to these insights, efforts are also being made in collaboration with CI stakeholders to define use cases where this technology provides a unique added value. As demonstrated in section 3.4.3 of this document, which presents a real application example of the 3D virtualization module, the tool allows for a detailed visual inspection of a specific moment in time. Users can freely navigate through the scene to focus the inspection on desired elements and take measurements of scene components.

3.3.4 Anonymization

Unlike the other modules, the anonymization tool described in this section of the document was not previously included in D7.2[42]. Its inclusion is motivated by the need to avoid issues with GDPR regulations, preventing the collection and dissemination of personal data through the inspection system, specifically, the potential appearance of faces in the captured images.

The main component of this solution is the AI face detection model, and a thorough review of the state of the art has been conducted to implement the best possible solution. The outcome of this review indicates that the model with the best average precision in the main benchmarks for this task is TinaFace [33], achieving an AP of 0.97 on the WIDER Face dataset (easy), 0.963 on WIDER Face (medium), and 0.934 on WIDER Face (hard). Figure 16 displays the model TinaFace (RestNet-50) as #rank1 in the benchmark [35].

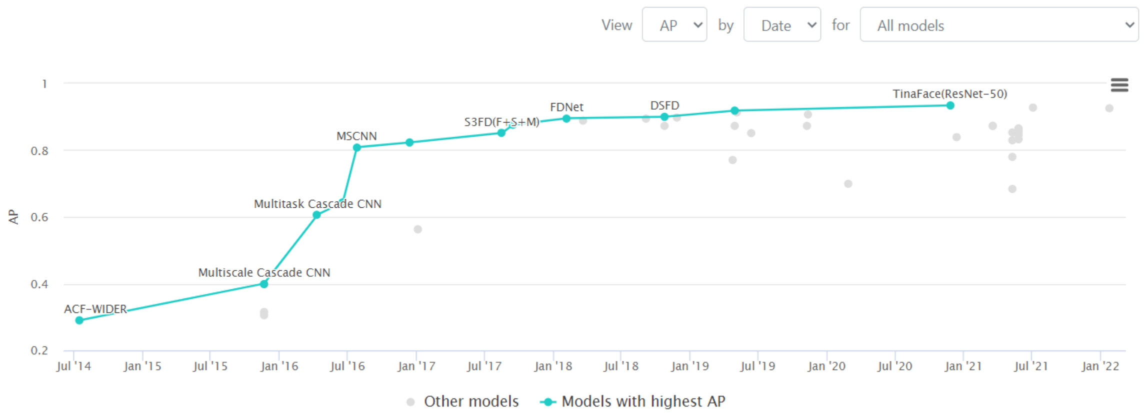


Figure 16: Face detection models leaderboard on WIDER Face (Hard). Source: [35].

In addition to its high performance, the simple architecture based on the deep convolutional neural network (DCNN) model, ResNet-50, makes it ideal for real-time applications due to its low inference time. For these reasons, the solution developed for Sunrise utilizes the open-source TinaFace implementation [36], including additional functionalities required to anonymize faces and to easily consume the service through an API request specifying the image to be anonymized.

The currently deployed API exclusively allows for the anonymization of faces, with notable performance as has been demonstrated. However, following discussions with CI stakeholders and partners, the possibility has been raised to add algorithms for detecting vehicle license plates or even full bodies. This could be particularly relevant in scenarios such as inspecting pipeline sections on ACO's beach facilities, where sensitive images of beachgoers could be captured despite face anonymization. The conclusion drawn from these discussions is that a significant incidence of these situations is not anticipated. Therefore, the decision on whether this is a real necessity has been deferred to later periods after conducting field tests to see if such cases occur and need to be addressed. The introduction of these license plates and full-body anonymization algorithms is considered technically feasible with relatively low additional effort and will proceed if deemed appropriate.

3.4 Tool modules lab validation

In the initial phases of the SUNRISE project, as detailed in D7.2[42], the critical need to validate the UAV platform and the proposed workflows within actual critical infrastructure settings was emphasized. These validations are slated for Pilots 1 and 2 in June 2024 and 2025, respectively. Currently, in the preparatory phase, Pilot 0, involving laboratory tests, the project is fine-tuning the solutions introduced in section 3.3 to ensure they are effective in real-world inspection scenarios.

While the real-world piloting validations are scheduled for future phases, this section aims to provide a brief analysis of the preliminary results. It highlights the practical relevance of these findings and sketches out the forthcoming steps for on-ground implementation and assessment. To achieve this, data from public sources are used, as well as some images that have been collected thanks to the collaboration of CI stakeholders.

As a visual introduction, and to showcase some of the outcomes in the most dynamic and striking way, it is recommended to watch the video presented at the Security Research Event 2023, organized by the European Commission, where SUNRISE participated as speaker. The video, which introduces the UAV inspection tool, can be found in the project's official Youtube channel, at the provided link: [34]. The video shows the state of development available in November 2023, since then there have been relevant developments, but it is still considered good information to facilitate the understanding of the tool's capabilities.

3.4.1 Object Detection and Semantic Segmentation

To illustrate some of the results obtained with the new models deployed in the detection and segmentation module, Figures 14, 15 and 16 present examples of real use cases where the solution has been successfully applied.

Firstly, Figure 17 demonstrates the performance of the YOLOv8x-seg model for crack detection and segmentation. This frame is extracted from a video taken at HDE facilities, and the detection of the visible crack is consistently maintained throughout the video, demonstrating the robustness of the algorithm. In addition to the performance of the task, the execution times of the model were also recorded, achieving an average value of 120 ms, which translates to more than 8 FPS, allowing us to affirm that the solution operates in real-time.

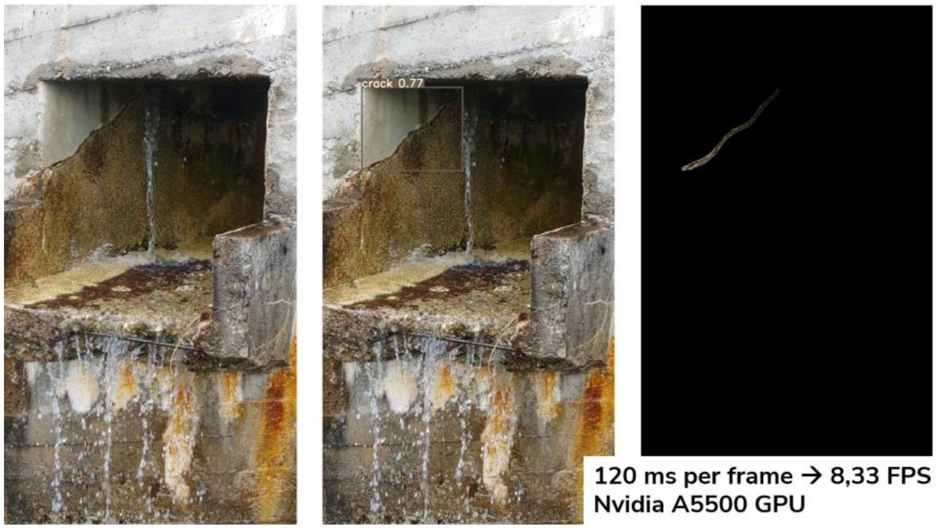


Figure 17: YOLOv8x-seg Crack detection and segmentation model inference examples on HDE premises.

Secondly, the example shown in Figure 18 demonstrates the ability of the YOLOv8x-seg model for corrosion detection and segmentation to highlight the problematic areas of a steel bridge structure, labeling each area with its respective degree of corrosion. The footage used in this test comes from the test set of [22].

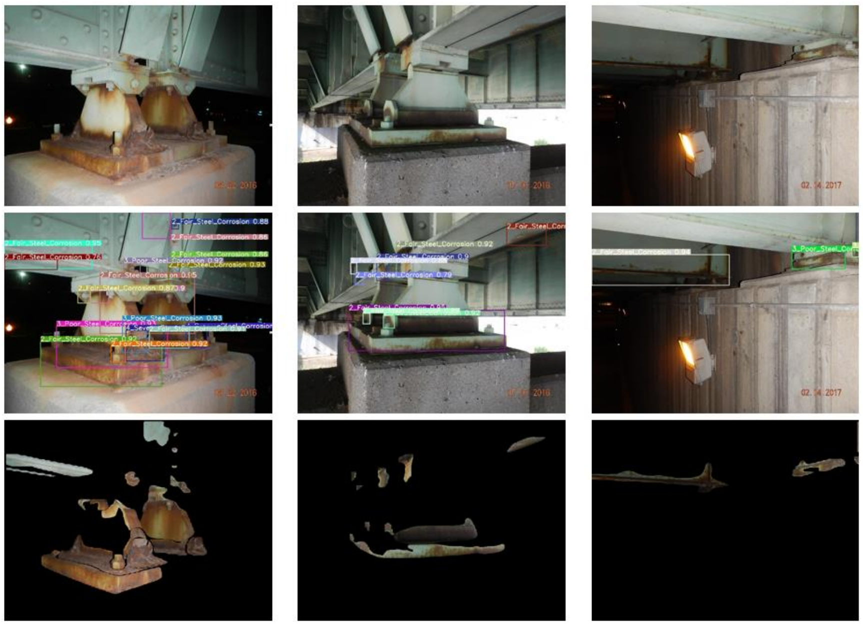


Figure 18: YOLOv8x-seg Corrosion detection and segmentation model inference examples on test set in dataset [22].

Third and lastly, an example of the use of the YOLO-World model is presented, which aims to demonstrate the versatility of this open-vocabulary and zero-shot solution set. For this purpose, a representative image of a real use case has been obtained, in which a tree is blocking a train track. As seen in Figure 19, the model is capable of understanding the concept introduced as a prompt for detection, in this case "tree in the railway", and of detecting it successfully, distinguishing it from the other trees clearly visible in the scene. The execution time of the complete detection pipeline does not exceed 0.45 seconds, so it could be used in inspection tasks considered near-real time.

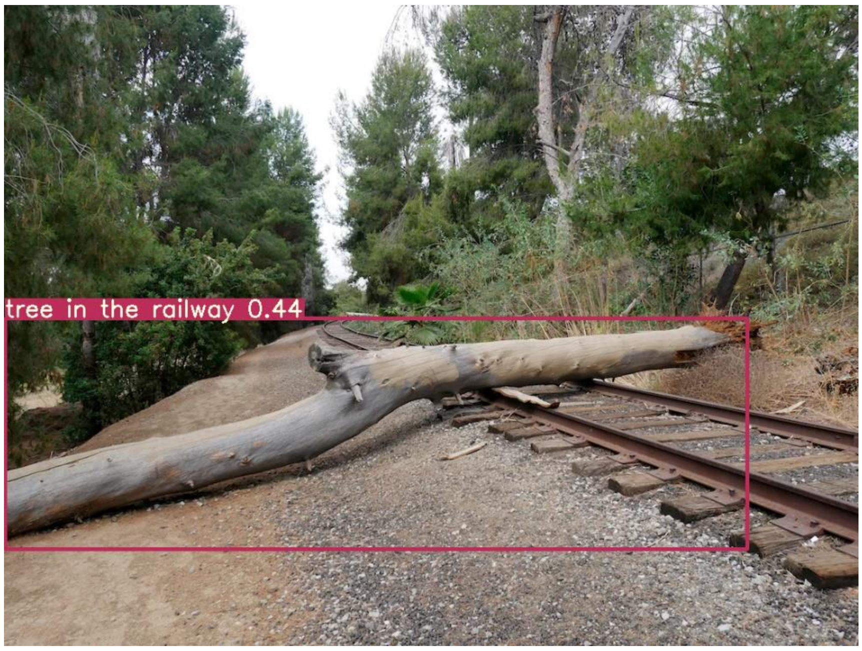
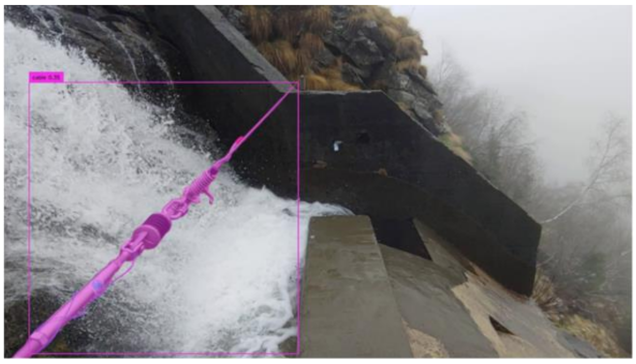


Figure 19: YOLO-World open-vocabulary example. Source: https://fullertonobserver.com/.

3.4.2 VQA

As outlined in section 3.3.2, the VQA module is perhaps the one with the highest degree of innovation and potential to solve a wide spectrum of inspection problems with a single tool. To exemplify the concepts introduced in the theoretical introduction of the V* and LLaVA1.6 models, and to further validate their use in real scenarios of CI stakeholders, two extra scenarios are attached, one related to the application of V*, and one with LLaVA1.6.

For the first scenario, the same methodology outlined in the use case of Figure 13 is followed, but this time the object to detect and analyze is a safety rope located next to a waterfall at the HDE facilities, in order to secure workers who have to clear the grate and perform maintenance tasks. On this occasion, Figure 20 does not include the entire prompt introduced to the SEAL VQA model; instead, it only shows the segmented image with the detection of the rope and the question asked by the CI operator, along with the model's response.

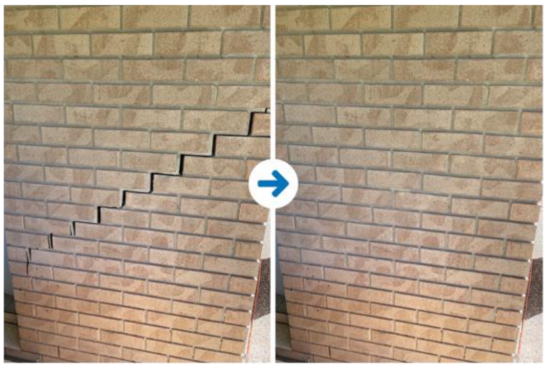


```
"describes in detail the condition of the object cable": "The image depicts a
rope or cable that is sitting atop a rocky, choppy body of water, possibly a
river with crashing waves. The cable appears to be hanging over the edge of a
concrete structure, which could be a bridge or some other type of water
infrastructure. The combination of the rope or cable and the rough waters
suggests that it might be used for mooring or securing purposes, or possibly for
maintaining the structure's stability. The scene conveys a sense of nature's raw
power, as the strong waves and currents could pose risks to those who come into
contact with the water.",
"is the cable firmly attached?": "Yes, the cable is firmly attached.",
"how is the water flowing?": "The water is flowing over a dam, creating a strong
current in the river below."
```

Figure 20: Safety cable status inspection in HDE facilities with Focus VQA module. Resume version, only user inspection questions (red) and model's answers (black) included.

As can be read in the image, the level of scene comprehension by the model is remarkable, clearly answering the question of whether the cable is firmly attached and providing a very solid interpretation of its possible use.

For the second example included, related to the validation of the applicability of LLaVA1.6 in inspection tasks, an image has been selected that shows the before and after of the repair of a structure or brick wall. On this occasion, as in the previous case, the model receives generic instructions on how it should behave, omitted in the image, along with the specific request of the operator, which can be read in red in Figure 21. The response of the model is highlighted in blue.



cracked-wall-beforeafter.jpg:`Describes the differences in the condition of structural elements and infrastructure` / **RESPOND:**`The image shows a side-by-side comparison of a brick wall with a visible crack or separation between the bricks. The crack is more pronounced on the left side of the image, indicating a potential issue with the structural integrity of the wall. The right side of the image shows the same wall with no visible signs of the crack, suggesting that the crack has been repaired or the wall has been restored.`

Figure 21: Example of change detection using LLaVA 1.6. Base image source: https://resinject.com.au/wall-repair/.

Once again, the deep level of comprehension of the model, in this case LLaVA 1.6, is reflected in the answer, not only being able to detect the crack in the left image but also being able to interpret that the wall has been repaired by comparing the images. These results confirm the potential of the solution to address various scenarios inspections, including change detection tasks between two different temporal moments.

In addition to the examples included in this section, this module has been tested with good results for other scenarios, such as the detection of clogged grates, open manholes, pipe leaks, and stones on the roof. Examples of these other scenarios have not been included in order to avoid extending the length of the document, considering that the functionalities of the tool are satisfactorily reflected with the included examples.

3.4.3 3D Virtualization

To validate this module, videos from various Point of Interest (POI) flights over a dam facility managed by partner HDE has been successfully acquired and used to reconstruct the scene in 3D. The full 3D reconstruction process is illustrated in the Figure 22.

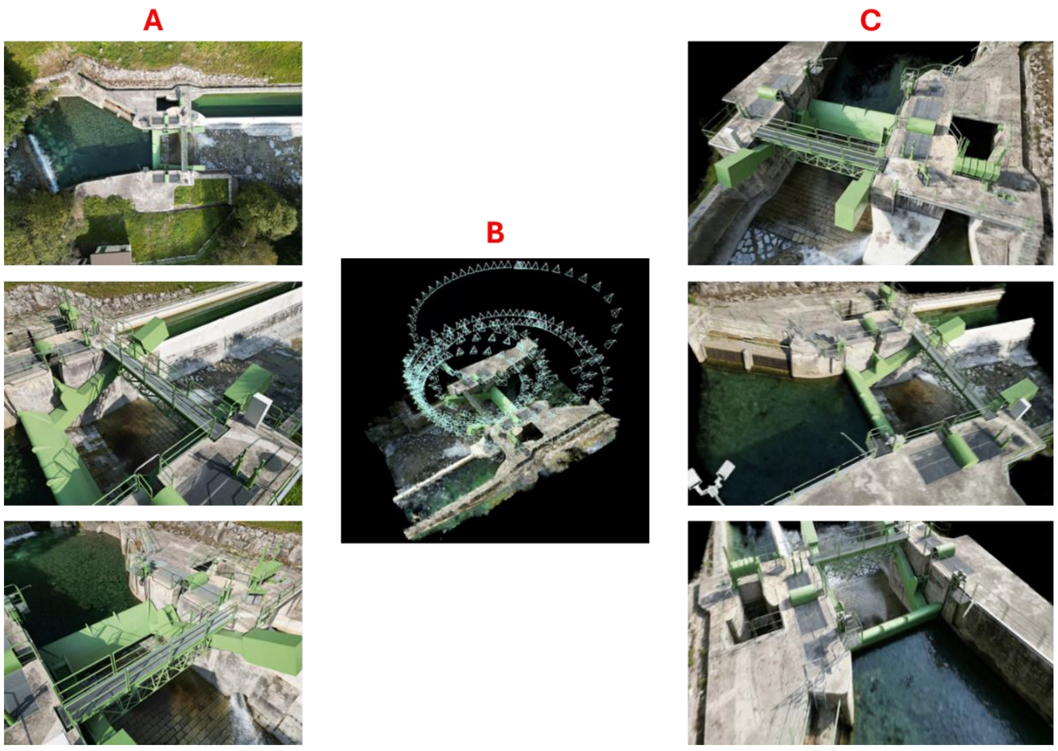


Figure 22: 3D virtualization HDE's dam reconstruction. (A) images extracted from raw POI videos; (B) camera path reconstruction; (C) output 3D rendering video frames.

The obtained model allows for the export of a rendering of the scenario in which a visual inspection process can be conducted on demand, as well as a 3D mesh file, Figure 23, on which measurements of distances or objects can be taken. Two different measurements on the 3D mesh have been carried out to validate the correlation between virtual and actual measurements. Measurement A corresponds to the 8-meter length of the dam bridge, and measurement B measures the 1.5-meter height of the stairs on the right side of the image.

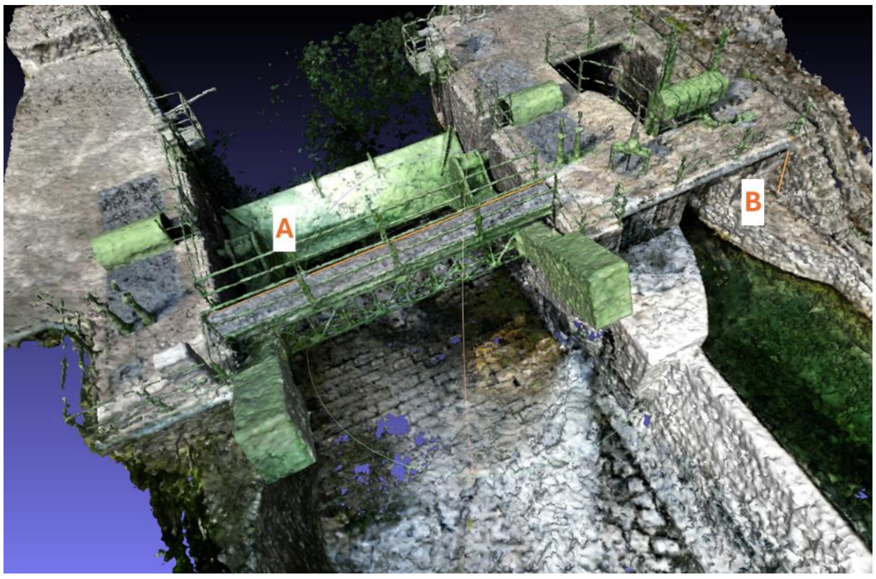


Figure 23: HDE dam 3D mesh measurement estimation. Orange line (A) represents bridge longitude measurement; Orange line (B) represents stair height measurement.

The data presented in Table 4 yield two very similar conversion values from 3DMesh to reality, with 3.52 for the bridge and 3.76 for the stairs. This consistency of the reconstruction allows for accurate measurements within the 3D model, with an error margin of approximately 5%. This margin could likely be attributed to the selection error of start and end points in the 3D model using the MeshLab tool.

Table 4: 3D mesh distance measurements. GT vs 3D model estimation.

Measured concept	Ground truth (m)	Estimation
(A) Bridge longitude	8.00	2.270
(B) Stair height	1.75	0.465

Lastly, as mentioned in the theoretical introduction of this tool, one of the challenges that need to be addressed is the size of the generated files. For instance, the file from which the screenshot in Figure 23 was taken is 32 MB, after having applied surface optimization and a reduced mesh density, attempting to minimize the file size and find the right balance between these parameters and the quality of the 3D model.

3.4.4 Anonymization

In reference to the anonymization solution, various images have been collected from the Critical Infrastructure stakeholders' premises featuring identifiable personnel. Figure 24 illustrates the outcome after applying the developed pipeline, which effectively removes personal data from images across diverse scenarios such as distant shots, side profiles, partial views, obscured faces, and close-ups.

Figure 24: Anonymized images taken in ACO and HDE facilities. Simple and complex scenarios, close-up, partially occluded, far away, or rotated faces.

3.5 Deployment

Building on the content introduced in Section 3.2 of this document and the foundations laid out in D7.2[42], this section delves into the deployment of the software tool currently in development, outlining two principal strategies: deployment as REST API services and as software integrated into the UAV's onboard Jetson card. The selection between these approaches will be guided by the unique operational demands and technical specifications of each use case.

The deployment hardware, referenced in Figure 6, can be any computer that satisfies specific minimum criteria, notably a GPU with 16GB of vRAM. These baseline requirements are readily achievable, and the containerization of all code facilitates straightforward deployment, whether on a collective scale within the project's framework or on an individual basis within the infrastructure of Critical Infrastructure (CI) stakeholders. Figure 25 illustrates the two previously mentioned types of computing hardware currently available. On one hand, there is the Edge option, which incorporates the 32GB Jetson Orin AGX module mounted on the UAV via the X2230D Carrier Board. On the other hand, the remote computing option is presented, utilizing an API deployed on a PC or on the 32GB Jetson Orin AGX Developer Kit to manage inspection requests. For further details about the capabilities and characteristics of these HW components, please refer to section 3.6 of document D7.2[42].

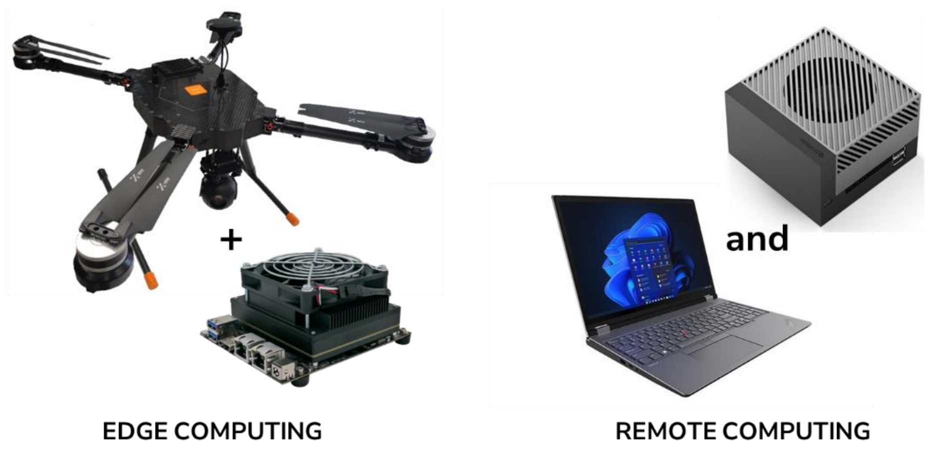


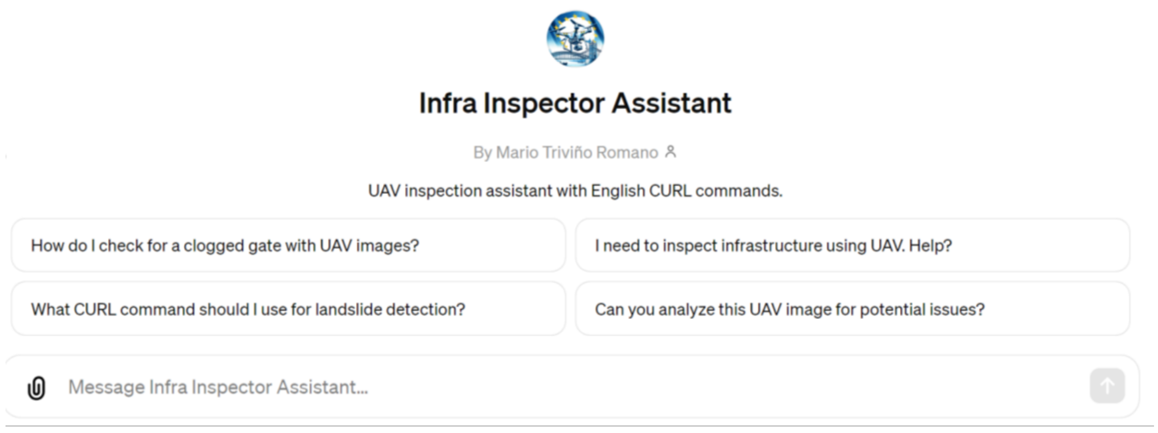
Figure 25: Types of computing hardware available in the UAV solution.

Presently, the APIs are operational on a Linux system equipped with a 12th Gen Intel® Core™ i9-12950HX processor × 24 and an Nvidia A5500 graphics card with 16GB of VRAM. Initiatives are also in progress to implement cloud solutions on the Jetson Orin AGX 32GB Developer Kit, acquired to test embedded solutions on the UAV. The main API's real-time models are already in place on the Jetson Orin, aiding in the facilitation of real-time edge processing trials during the forthcoming pilot phase. Successful tests have been conducted on the internal communications between the various deployed components, as well as communication with the GUI delineated in Section 4.

Once the solutions are deployed, the focus shifts to trying to make them as easy as possible for operators to use. The capabilities of open-vocabulary models and broad-spectrum tools integrated enable the use of the tool for a vast array of distinct inspection scenarios. However, this aspect introduces one of the main challenges associated with this deployment: the necessity to instruct the UAV inspection tool on the specific type of inspection and the precise settings to determine which processes from the extensive range available will be executed in each instance.

For instance, if the CI operator needs to ascertain whether a waterfall's grate is clogged, they must inform the tool of this objective and define the parameters to be used (or select this pre-recorded routine from a list). This ensures the tool accurately employs the modules, utilizing, for example, pertinent questions for that scenario in the VQA module (e.g., Figure 26 displays the initial interface of this virtual assistant specialized in utilizing the UAV inspection API."Is the grate clogged?") or relevant concepts in the detection/segmentation module (e.g., “grate”).

To simplify the interaction process with the deployed API for users, particularly aiding Critical Infrastructure (CI) operators tasked with utilizing the UAV remote inspection tool to do so more intuitively and without extensive training, a specialized ChatGPT has been fine-tuned. Figure 26 displays the initial interface of this virtual assistant specialized in utilizing the UAV inspection API.


Infra Inspector Assistant

By Mario Triviño Romano

UAV inspection assistant with English CURL commands.

How do I check for a clogged gate with UAV images?

I need to inspect infrastructure using UAV. Help?

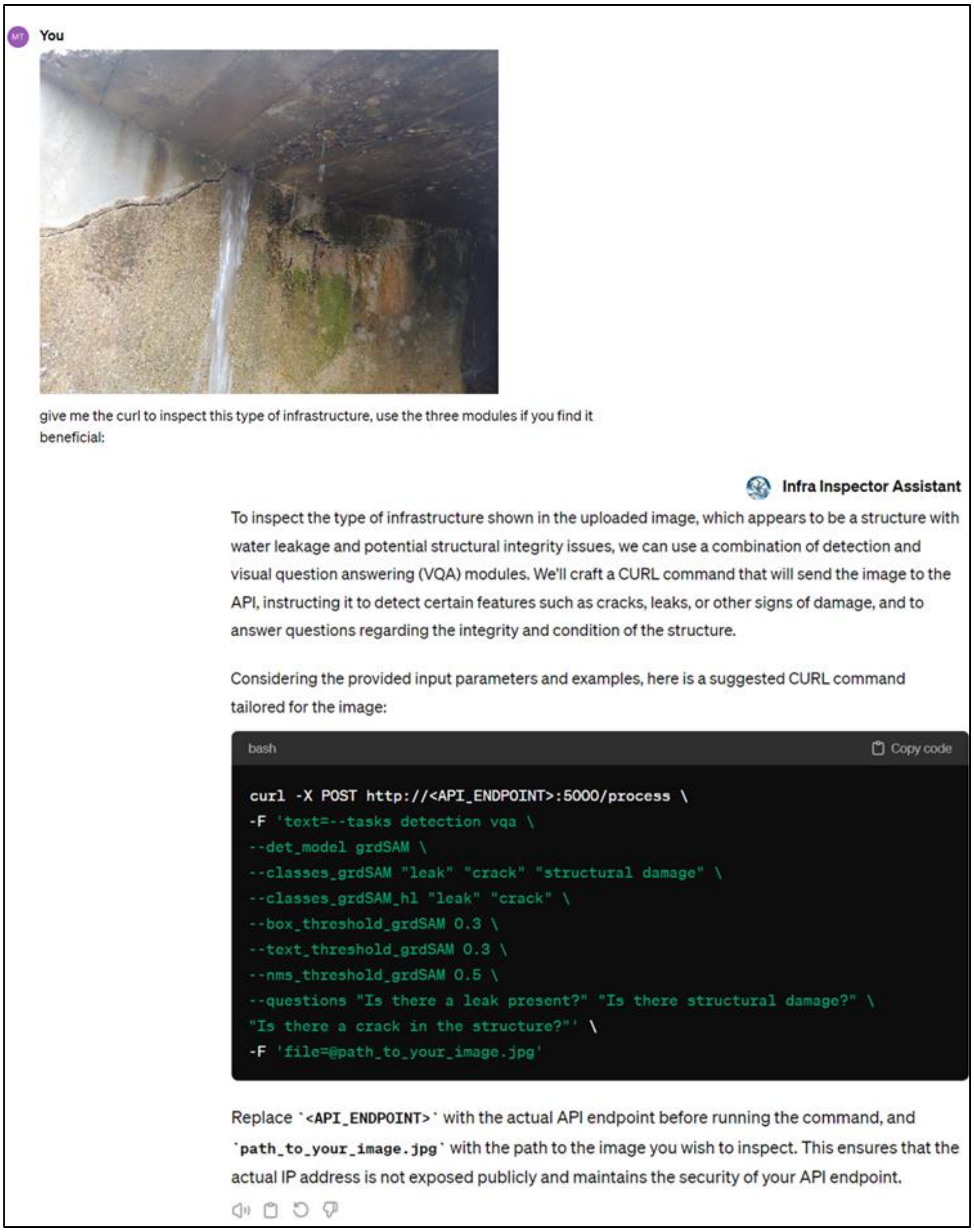
What CURL command should I use for landslide detection?

Can you analyze this UAV image for potential issues?

Message Infra Inspector Assistant...

Figure 26: GPT-4 Infra Inspector Assistant chatbot graphical user interface.

This version is customized based on OpenAI's GPT-4 model [37]. The resulting chatbot is familiar with all possible input parameters that the API can accept and has learned the appropriate CURL commands to provide to the user, based on the inspection task at hand, through a dozen examples. This virtual assistant is capable of advising users on which modules to utilize and which models within those modules best suit their tasks. It can even frame relevant questions for the VQA module or identify the concepts to detect/segment. Additionally, leveraging GPT-4's ability to interpret images, users can submit a representative scene of their specific problem and inquire with the assistant for an API request that facilitates the inspection of elements within that image. For illustrative purposes, Figure 27 provides an example where this user-assistant interaction is depicted, using an image as input, though a textual description of the intended inspection could also have been used.



You

give me the curl to inspect this type of infrastructure, use the three modules if you find it beneficial:

Infra Inspector Assistant

To inspect the type of infrastructure shown in the uploaded image, which appears to be a structure with water leakage and potential structural integrity issues, we can use a combination of detection and visual question answering (VQA) modules. We'll craft a CURL command that will send the image to the API, instructing it to detect certain features such as cracks, leaks, or other signs of damage, and to answer questions regarding the integrity and condition of the structure.

Considering the provided input parameters and examples, here is a suggested CURL command tailored for the image:

```bash
curl -X POST http://<API_ENDPOINT>:5000/process \
-F 'text=--tasks detection vqa \
--det_model grdSAM \
--classes_grdSAM "leak" "crack" "structural damage" \
--classes_grdSAM_hl "leak" "crack" \
--box_threshold_grdSAM 0.3 \
--text_threshold_grdSAM 0.3 \
--nms_threshold_grdSAM 0.5 \
--questions "Is there a leak present?" "Is there structural damage?" \
"Is there a crack in the structure?"' \
-F 'file=@path_to_your_image.jpg'
```

Replace `<API_ENDPOINT>` with the actual API endpoint before running the command, and `path_to_your_image.jpg` with the path to the image you wish to inspect. This ensures that the actual IP address is not exposed publicly and maintains the security of your API endpoint.

Figure 27: GPT-4 Infra Inspector Assistant chatbot example.

As observed in the dialogue between the user and the virtual assistant in Figure 27, the virtual assistant's understanding of the scene depicted in the image is very high, resulting in completely valid information (tested) regarding the optimal configuration and parameters to include in the API request. Both the structure of the generated CURL and its content are suitable for carrying out that inspection, even though this was not one of the examples included in the small training dataset used to teach the assistant.

The advantage of using this custom ChatGPT is reflected in the outcomes; the best possible results can be achieved by employing the most cutting-edge MLLM model in the current state of the art, GPT-4. The downside is that this custom GPT functionality is only available to paying users of the ChatGPT platform, which limits its use to those without a subscription to this service. One potential way to mitigate this drawback is to replicate the fine-tuning process with an open-source MLLM or LLM, such as LLaMA 1.6 or Mixtral, for example. However, the quality of the results would be lower, and a much larger number of examples would be required, which is a significant obstacle, as creating such datasets can be very time-consuming.

For all these reasons, the current idea is to opt for this solution with custom ChatGPT. Although it may not be completely open source, it is the most technologically advanced and best able to assist users in maximizing the deployed tools.

Regardless of the specific tool described previously, API users will be provided with the necessary documentation to understand all existing models and parameters. A first version of this user manual is provided in Annex 1. This will enable users to effectively utilize the solutions without reliance on the virtual assistant. Furthermore, efforts will be made to define a series of presets with common inspection scenarios, allowing users to launch predefined routines such as crack inspection, real-time fire monitoring, pipe leak detection, ceramic insulator breakages, clogged grates, etc.

With the integration testing conducted to date, and all APIs deployed and accessible through a private VPN established using ZeroTier [38], there is a favorable position for commencing the first pilot phase. The outcomes of this phase will be detailed in the upcoming deliverable of WP7.

3.6 UAV platform integration

Following advancements in technology since the initial proposal of the aerial vehicle, new options have emerged. Consequently, a decision was made to substitute the originally designated UAV aerial vehicle specified in deliverable D7.2[42] with the MERA EXT UAS edition, while maintaining the overall UAV platform architecture unchanged. This choice was influenced by the superior portability of the MERA EXT UAS edition, characterized by a more compact design that enhances ease of transportation and deployment during field operations.

3.6.1 Aerial Vehicle Hardware Specifications

A more suitable UAV for deployment in SUNRISE project is the "MERA EXT UAS edition", as depicted in Figure 28. This UAV system is thoroughly crafted to offer an unmatched flight range, effortless integration of payloads, improved reliability, and streamlined handling, ultimately cutting down on packing and deployment time. MERA is a model aircraft specifically engineered to accommodate First Person View (FPV) equipment, ideal for tasks like remote sensing and low-altitude aerial photography/mapping. Its innovative design highlights a modular composite structure, quick assembly, and a detachable payload bay as key distinguishing features. Mera UAV has been factory designed to support the Jetson Orin microcomputer, considering the UAV's aerodynamic characteristics and the spatial constraints within its primary hull.

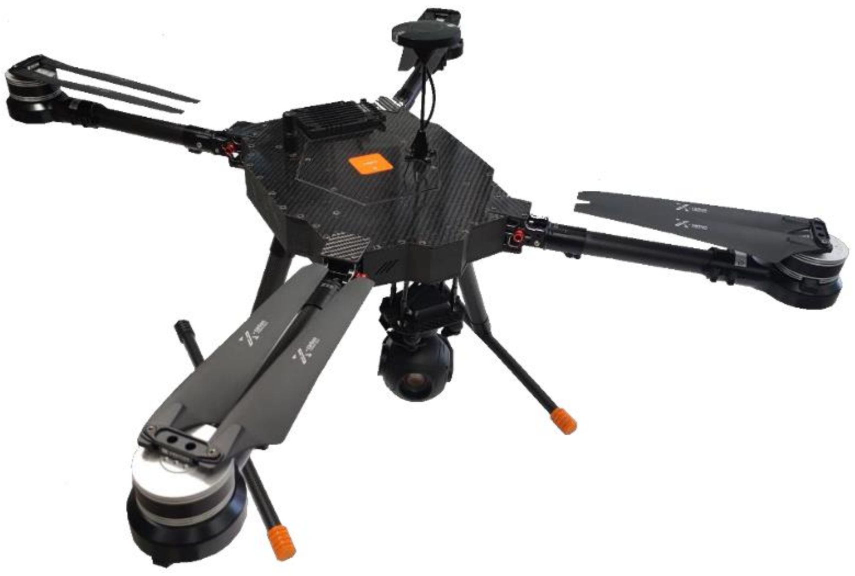


Figure 28: Aerial Vehicle MERA EXT UAS.

Table 5. UAV Specs.

UAV Specifications	Performance Metrics
Maximum Flight Speed	76 km/h
Maximum Flight Altitude	10.000 ft
Maximum Flight time (incl. payload, battery)	46 min
Maximum Wind Resistance	19 m/s

UAV Specifications	Performance Metrics
Maximum Transmission Distance (Line of Sight – LOS)	10 – 20 km
Type	Multicopter x4 with foldable arms
Minimum number of operators	1 operator
Landing Equipment	Carbon Landing Skids
Arms	x4 foldable/ unfoldable motor arms
Dimensions (unfolded, without propellers)	970mm
Length	740mm
Width	760mm
Height	530mm
Maximum Take Off Weight (MTOW)	7500g
Fuselage Ingression Rate	Designed for IP54
Precipitation	Light Rain (2 mm/h)
Operational Temperature	-20° to +60° C
Relative Humidity (ground)	0 to 80% (without condensation)

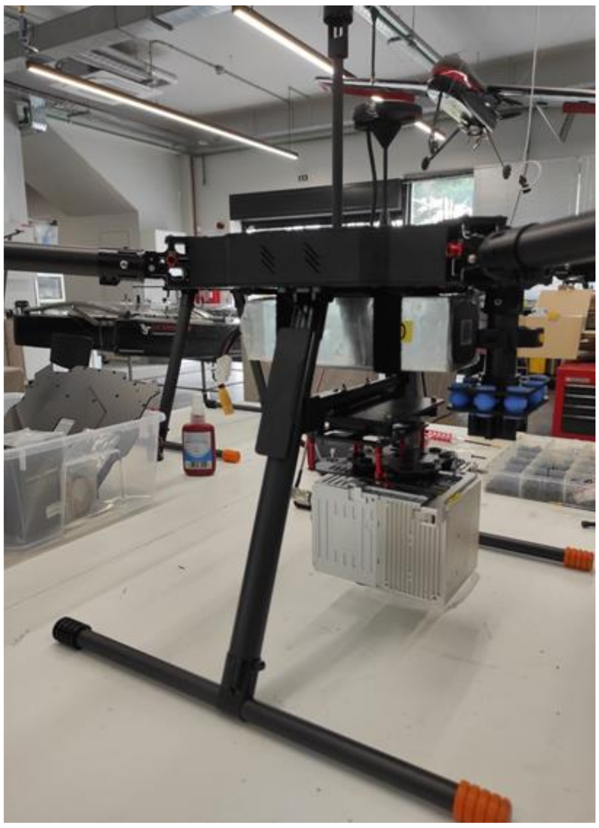


Figure 29: Aerial Vehicle MERA EXT UAS with payload.

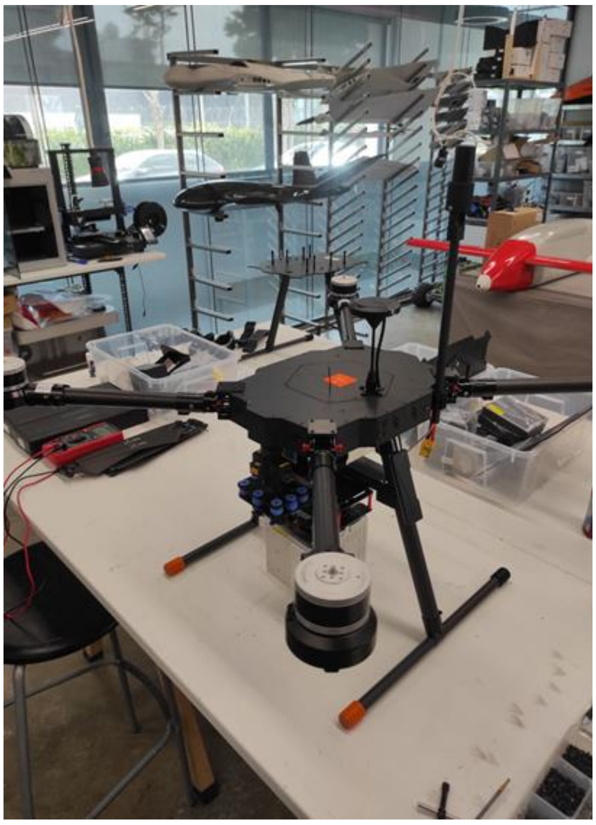


Figure 30: Aerial Vehicle MERA EXT UAS with payload.

3.6.2 Internal Units' Connections and Communications

In Figure 31 is shown the interconnection schematic of the internal modules within the UAV platform.

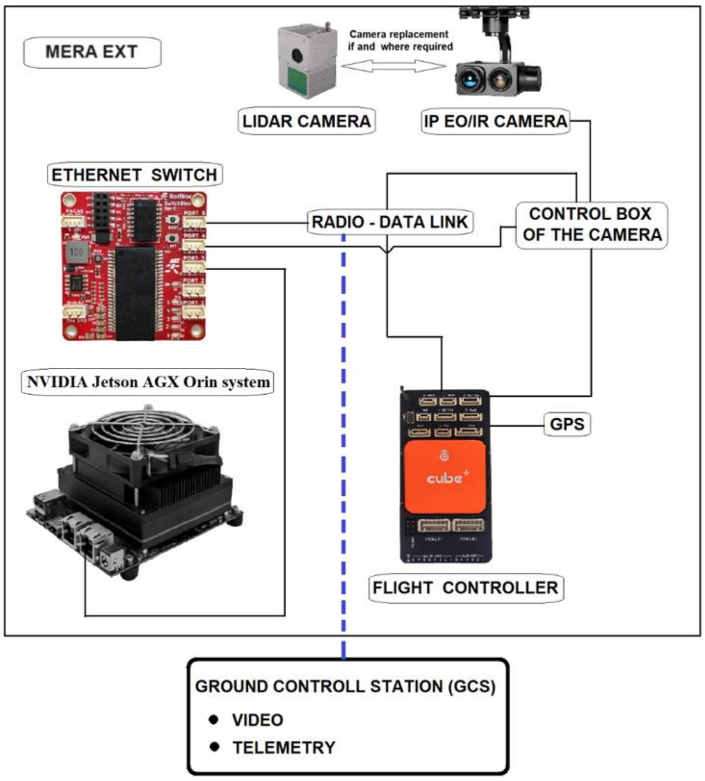

MERA EXT
Camera replacement if and where required
LIDAR CAMERA
IP EO/IR CAMERA
ETHERNET SWITCH
RADIO - DATA LINK
CONTROL BOX OF THE CAMERA
NVIDIA Jetson AGX Orin system
GPS
FLIGHT CONTROLLER
GROUND CONTROLL STATION (GCS)
VIDEO
TELEMETRY

Figure 31: Inspection Tool: “Internal Connections Diagram and Communication”.

The connections and communications remain unchanged from the connectivity diagram outlined in the preceding deliverable D7.2[42]. The content remains unchanged from the previous document, but it is illustrated again with the corresponding UAV model specified.

3.6.3 Relay Drone System

In our pilot execution, our primary emphasis lies on practical considerations. This includes obtaining the required permits, determining the feasible actions within the critical infrastructure operators' constraints, and adhering to the frequently restrictive regulations governing UAV flights in critical infrastructure areas. Our focus is directed towards the inspection technology itself, rather than delving into the intricacies of operations such as Beyond Line of Sight (BLOS) flying. The modifications in the pilot procedures are outlined in Section 5, detailing any deviations from the initial plans and highlighting that no relay drone system is required.

4 User interface for remote infrastructure inspection

This section of the document provides an update on the user interface dashboard dedicated to inspecting the critical infrastructure events captured from UAV or satellite, continuing the work from task T7.2. Maintaining the structure used in the previous deliverable, this section goes deeper to the details of the integration tools.

4.1 General context

The design and architecture of the Dashboard User Interface remains the same as described in the last deliverable and its primary goal continues to be delivering real-time images to users, showcasing areas/components/points of failure in critical infrastructure, including damaged components, structural issues, corrosion, vegetation obstruction, and more.

The Backend Coordinator, a tool that was previously pending implementation, has now been successfully completed. Revisiting the document D7.2[42], this component is responsible for handling incoming messages and events, fetching data from the front of the MQTT queue, transmitting live or historical data to the Dashboard UI for visualization, fulfilling historical data requests from the Dashboard UI, communicating with the Reporting Subsystem to gather and process data, and managing outbound data transmission by placing it in the appropriate MQTT queue.

After the CI operators' input, a minor change is implemented to the login page by incorporating a Two-Factor Authentication (2FA) system for enhanced security. This additional layer of security will require users to provide not only their usual login credentials (such as username and password) but also a secondary piece of information that only they have access to, typically a unique code sent to their mobile device or generated by an authenticator app. By introducing two-factor authentication, the login process becomes more secure, significantly reducing the risk of unauthorized access and enhancing overall account protection.

4.2 Architecture: high level Implementation

To enhance clarity, it is essential to revisit the architectural framework underlying the Dashboard UI development. This will enable readers to swiftly grasp the content depicted in Figure 32, despite its detailed description in deliverable D7.2[42], section 4.2. The selected web application architecture is described through the numbered bus lines as follows:

1. Incoming messages/events from UAV/Satellite systems are received. All this data is routed through an MQTT bus system. Within this system, the data is systematically queued, ensuring a sequential flow.

2. The Backend Coordinator processes all incoming messages/events. It retrieves the data at the front of the MQTT queue.

3. All incoming messages/events are internally stored in the Backend Inventory (MongoDB server).

4. The Backend Coordinator sends live or historical data to the Dashboard UI for visualization and responds to historical data requests from the Dashboard UI.

5. The Dashboard UI communicates with the Google Maps infrastructure to render maps, markers, points of interest, and heat maps, among other elements.

6. The Backend Coordinator sends requests to the Reporting Subsystem in order to compile the requested data and then receives the results.

7. The Reporting Subsystem and the Backend Inventory communicate with each other in order to process the requests, and subsequently transmits the results to the Backend.

8. The Dashboard UI obtains an Access Token from the Identity Server to access backend APIs. Access to the UI is exclusively granted to authorized users, with authentication and authorization handled by a dedicated Authentication/Authorization unit, responsible for controlling user access and logging into the application.

9. Additional public services can offer crucial meteorological data, weather forecasts, maritime information, alerts, and more for visualization within the Dashboard UI.

It is important to note that the connection between the Backend Coordinator and the MQTT system is bidirectional. If any data needs to be transmitted from the application outward, the Backend Coordinator places it in MQTT, within the corresponding queue.

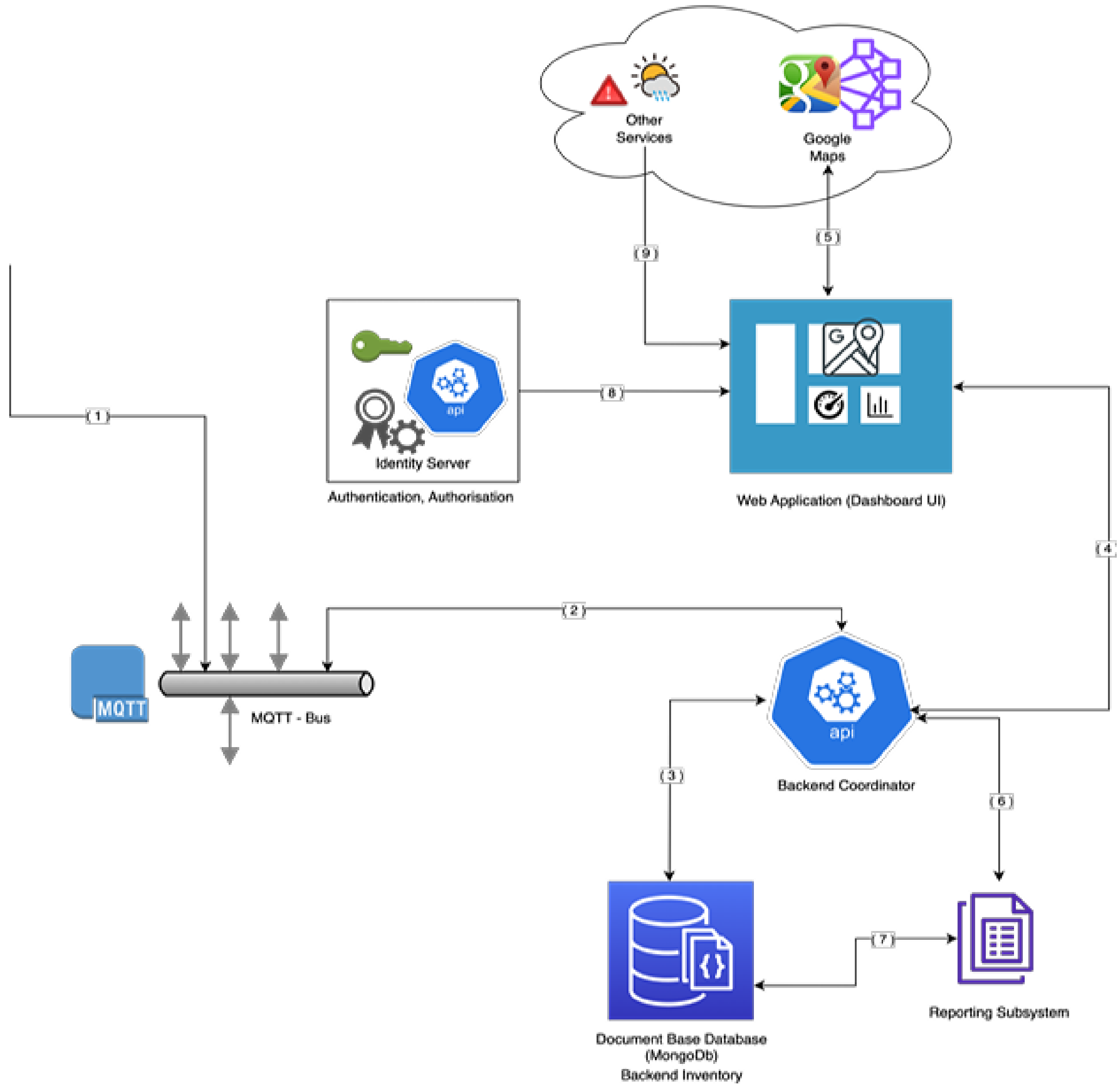


Figure 32: UI Architecture Diagram.

4.2.1 Internal Components - Backend Coordinator

No changes are tracked on the dashboard UI internal components. The most recent work update has been completed on the backend coordinator, where the following services have been implemented:

MongoDbService. This background hosting service is responsible for writing all events received from the corresponding Detection Services to the "UAV" and "SAT" collections.

MqttSubscriberService. This background hosting service is responsible for listening to the MQTT topics and forwarding these messages to the Dashboard UI using WebSocket communication for real-time event presentation. It utilizes the MongoDbService to store these messages in the Document-Based Database (MongoDB) Backend Inventory.

ReportingService. This background hosting service is tasked with generating filters based on criteria selected by end users on the Dashboard. It listens to user requests from the Dashboard for searching and reporting purposes. Upon receiving these requests, it executes queries on the Document-Based Database (MongoDB) Backend Inventory through the MongoDbService of the Backend Coordinator Service. Subsequently, it returns the results to the Dashboard Web App for further visualization and exporting functionalities.

4.2.2 Database security

Our MongoDB implementation, exclusively accessed by the Backend Coordinator service, is fortified with a comprehensive array of security features to ensure the integrity, confidentiality, and availability of our data.

Authentication and Authorization: the Backend Coordinator service serves as the solitary user of our MongoDB databases, managing authentication and authorization processes. Authentication is enforced through a robust username/password mechanism, allowing the Backend Coordinator service to securely access MongoDB resources. Role-based access control (RBAC) is meticulously configured to grant the Backend Coordinator service granular permissions, restricting its access to databases and operations based on predefined roles.

Encryption: our MongoDB deployment, exclusively accessed by the Backend Coordinator service, implements encryption at rest and in transit to fortify data security. Data at rest is safeguarded using the WiredTiger encryption engine, employing AES-256 encryption to protect data files stored on disk. Encryption in transit, enforced through TLS/SSL protocols, ensures that data exchanged between the Backend Coordinator service and MongoDB servers remains encrypted during transmission, safeguarding it from unauthorized access and tampering.

Auditing and Logging: the Backend Coordinator service integrates robust auditing and logging functionalities, enabling comprehensive tracking and monitoring of all database activities. Audit logs meticulously record authentication attempts, database commands, and administrative actions initiated by the Backend Coordinator service, facilitating security analysis and compliance auditing.

Network Security: stringent network access controls are configured by the Backend Coordinator service to restrict access to MongoDB servers exclusively to authorized entities. Network encryption using TLS/SSL protocols is enforced to secure data transmission between the Backend Coordinator service and MongoDB servers, bolstering network security and mitigating the risk of unauthorized access.

Authentication Plugins: as the sole user of MongoDB, the Backend Coordinator service does not require integration with external authentication systems. Authentication is exclusively managed through the username/password mechanism.

Security Best Practices: our MongoDB deployment, managed by the Backend Coordinator service, adheres to industry-standard security best practices. Secure deployment configurations, access control policies, encryption settings, and auditing configurations are implemented to uphold the integrity and security of our MongoDB environment. Regular updates and patching are diligently performed to address security vulnerabilities and maintain the resilience of our MongoDB infrastructure.

4.2.3 Rest API protocol of Dashboard UI

The security of critical infrastructures is a top priority that requires our utmost attention. Given the importance of this topic, it is necessary to revisit the design considerations from the previous deliverable, D7.2[42]. This will ensure that the security aspects are thoroughly addressed and incorporated into the overall system design.

Authentication, Authorisation and Audit Logging component, is responsible for intelligently controlling access to UI tools system functions and interfaces (both GUI and REST-API), enforcing policies, and keeping an audit trail of events happening. Based on assigned roles, authenticated users are able to access different UI system functions and interfaces. The audit logging mechanism log several types of information that the system generates during normal execution, such as data changes and actions/commands invoked by the end-users.

Structuring the UI web application to support a security token service (Authentication, Authorization and Audit Logging component) leads to the architecture and protocols shown in Figure 33.

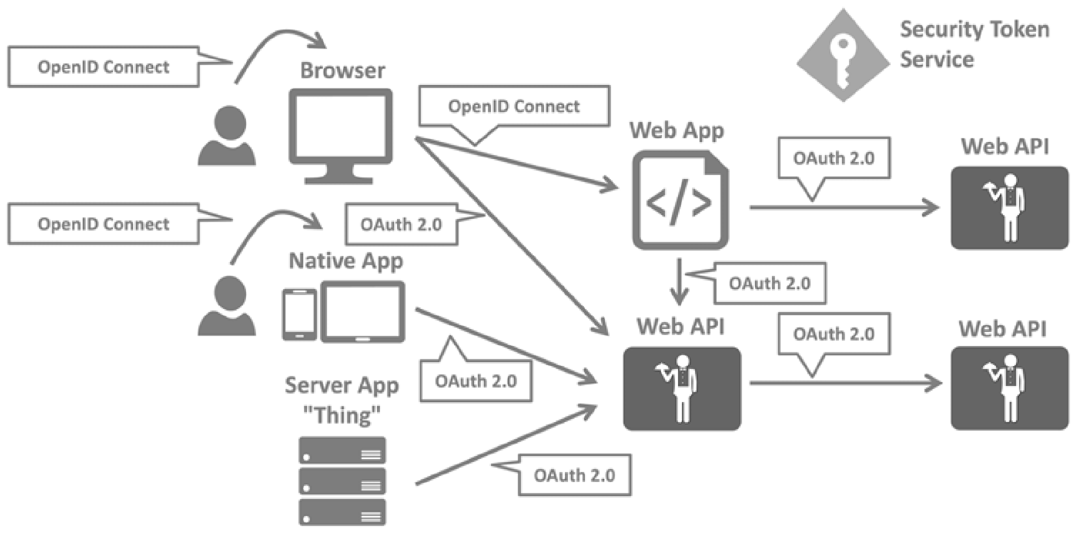


Figure 33: Security Token Architecture and Protocols.

The integration of OAuth-compliant REST API authentication and authorization between the Dashboard UI client and the Backend Coordinator b service represents a robust and secure framework that has been successfully implemented in our system. This implementation enhances security by ensuring secure authentication, fine-grained authorization, token-based security, secure communication, client credentials, token revocation, and adherence to standardization and best practices.

Secure authentication is achieved through the use of OAuth, which replaces the direct sharing of sensitive credentials like usernames and passwords with the issuance of access tokens. These tokens serve as proof of authentication and are included in subsequent API requests to the Backend Coordinator backend service, ensuring secure access to protected resources. The Identity Server issues access tokens during the OAuth authentication process, providing a secure and efficient way to authenticate users.

Fine-grained authorization is another key aspect of our OAuth implementation. It empowers users to grant specific permissions (scopes) to the Dashboard UI client, dictating the actions that the client can perform on behalf of the user. This ensures that only authorized operations are executed, as the Dashboard UI client requests access to specific scopes during the OAuth authentication process, which are validated by the authorization server (Identity Server). The resulting access tokens contain the appropriate permissions, limiting the client's access to only authorized resources and functionalities within the Backend Coordinator backend service.

Token-based security is a crucial aspect of our system, as it relies on access tokens for secure communication between the Dashboard UI client and the Backend Coordinator backend service. These tokens are short-lived and cryptographically signed, minimizing the risk of unauthorized access and data breaches. Access tokens serve as temporary credentials, granting access to protected resources for a limited duration. Upon expiration, the Dashboard UI client obtains new tokens through the OAuth authentication process, reducing the window of vulnerability and enhancing security.

Secure communication is enforced through the use of HTTPS (HTTP Secure), which encrypts data transmitted between the Dashboard UI client and the Backend Coordinator backend service. This encryption ensures the confidentiality and integrity of sensitive information, including access tokens and user data, mitigating the risk of eavesdropping and tampering. Our OAuth implementation also supports client credentials, allowing the Dashboard UI client to authenticate itself directly with the authorization server (Identity Server). This mechanism verifies the identity of the client, ensuring that only registered and trusted clients can access protected resources in the Backend Coordinator backend service.

Token revocation is another important feature of our OAuth implementation. It enables users to invalidate access tokens if unauthorized access or token compromise is suspected, providing a proactive approach to security that empowers users to mitigate risks associated with unauthorized access and data breaches. Finally, our OAuth implementation adheres to industry-standard protocols and best practices for authentication and authorization, ensuring interoperability, consistency, and adherence to industry security guidelines.

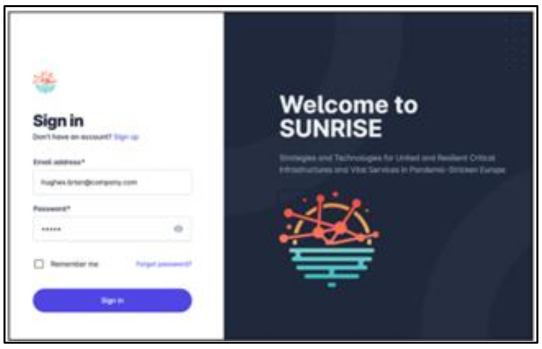
In summary, our OAuth-compliant REST API authentication and authorization implementation, utilized between the Dashboard UI client and the Backend Coordinator backend service, exemplifies a robust and secure framework that prioritizes user privacy, data security, and regulatory compliance.

4.2.4 Two-Factor Authenticator of Dashboard UI

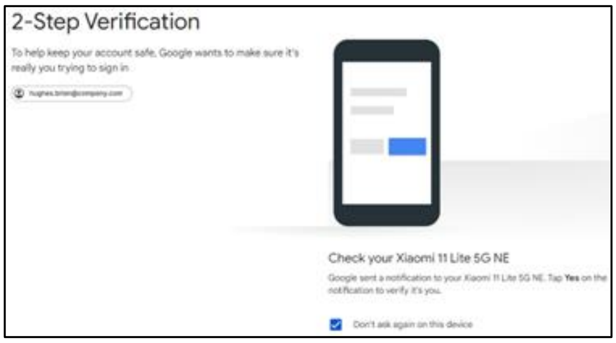
To enhance the security access of the Dashboard UI, a Two-Factor Authentication feature using Google Authenticator has been integrated with the existing Identity Server OAuth service. Two-factor authentication (2FA) on web applications works as follows: The user first logs in with their username and password, which represents the first authentication factor - something they know. After successfully entering the username and password, the web application then prompts the user to provide a second form of authentication, such as a one-time code sent to their registered mobile device. This one-time code represents the second authentication factor - something the user has.

The user receives the one-time code, typically via SMS or a mobile app, and enters it into the web application to complete the login process. Once the user provides the correct one-time code, they are granted access to the web application.

The key aspect of how 2FA works is that it requires two independent factors to authenticate the user - something they know (password) and something they have (mobile device). This provides an extra layer of security beyond just a username and password, making it much harder for an attacker to gain unauthorized access. Common 2FA methods include one-time codes sent via SMS, mobile app authenticators, and hardware security keys. Implementing 2FA is recommended by security experts to better protect user accounts and sensitive data on web applications.


Sign in
Don't have an account? Sign up
Email address*
Password*
Remember me
Forgot password?
Sign in
Welcome to SUNRISE
Strategies and Technologies for United and Resilient Critical Infrastructures and Vital Services in Pandemic-Stricken Europe

(1) Sign in


2-Step Verification
To help keep your account safe, Google wants to make sure it's really you trying to sign in
Check your Xiaomi 11 Lite 5G NE
Google sent a notification to your Xiaomi 11 Lite 5G NE. Tap Yes on the notification to verify it's you.
Don't ask again on this device

(2) Step Verification

Figure 34: Two-Factor Authentication feature.

Google Authenticator stands out as a preferred option for two-factor authentication (2FA) due to several key advantages. Its utilization of a time-based one-time password (TOTP) algorithm enhances security by generating unique, time-sensitive codes that are challenging for attackers to predict or intercept. This dynamic approach surpasses static codes or SMS-based methods, bolstering the overall security posture.

Moreover, the ease of use associated with Google Authenticator contributes to its popularity. Available as a free app on both Android and iOS platforms, its user-friendly interface and straightforward setup process streamline the user experience. By enabling users to effortlessly set up accounts through QR code scanning or manual key entry, Google Authenticator minimizes complexity and hardware requirements, fostering accessibility.

Additionally, Google Authenticator's offline functionality sets it apart from other 2FA solutions. Operating independently of internet connectivity, the app ensures reliability in scenarios with limited or no network access. This autonomy enhances user convenience and security, making it a dependable choice for safeguarding accounts across various platforms.

Furthermore, the widespread compatibility of Google Authenticator with numerous online services underscores its versatility. Embraced by major platforms like Google, Facebook, and Dropbox, its broad adoption enhances its applicability and convenience for users seeking consistent 2FA protection across diverse services.

The credibility of Google as a reputable company further reinforces the trustworthiness of Google Authenticator. Known for its commitment to security and privacy, Google's regular updates to address vulnerabilities and enhance functionality instill confidence in users, solidifying Google Authenticator as a reliable 2FA solution.

While Google Authenticator offers compelling benefits, it's essential to acknowledge the existence of alternative 2FA options like Authy, Microsoft Authenticator, or hardware tokens such as YubiKey. The selection of a 2FA method may hinge on factors like user preference, system compatibility, and specific security needs. Ultimately, the primary objective of 2FA remains to fortify security by mandating dual authentication, with Google Authenticator emerging as a popular and effective choice for achieving this objective.

4.2.5 WebSocket of Dashboard UI

In our system, real-time data presentation between the Backend Coordinator and the Dashboard UI is facilitated through SignalR, a library that implements the WebSocket protocol. This implementation prioritizes security to safeguard sensitive information exchanged in real-time. The following details highlight the security aspects of this setup:

Encrypted Communication: SignalR WebSocket connections are established over HTTPS, ensuring encrypted communication between the Backend Coordinator and the "Dashboard UI." This encryption mechanism utilizes SSL/TLS protocols to protect data from interception and tampering, maintaining the confidentiality and security of real-time information.

Same-Origin Policy (SOP): SignalR WebSocket connections adhere to the same-origin policy, restricting communication between scripts from different origins. By enforcing SOP, SignalR mitigates the risk of cross-site scripting (XSS) attacks, ensuring that real-time data remains isolated and secure within the application's origin.

Built-in Support for Secure Protocols: SignalR WebSocket communication supports secure protocols like TLS, enhancing data encryption, authentication, and integrity protection. Leveraging TLS ensures that WebSocket connections are secure, resistant to attacks, and safeguarded against eavesdropping and data tampering.

Authentication and Authorization: SignalR integrates with various authentication mechanisms to verify client identities and enforce access control policies. This enables the Backend Coordinator to authenticate users, control data access based on permissions, and implement secure authentication mechanisms tailored to the application's needs.

Server-Side Security Measures: The SignalR WebSocket server implements additional security measures such as input validation, rate limiting, and secure coding practices to protect against attacks and vulnerabilities. Proactive security measures at the server level ensure the integrity and robustness of real-time data transmission.

Cross-Origin Resource Sharing (CORS): SignalR WebSocket communication supports CORS, allowing the Backend Coordinator to specify trusted origins for accessing real-time data. By defining CORS policies, SignalR enhances security, prevents unauthorized cross-origin requests, and ensures WebSocket connections are established only from trusted sources.

Secure Deployment Configurations: Proper configuration and securing of the SignalR WebSocket server infrastructure are crucial for ensuring the overall security of real-time data transmission.

Implementing firewall rules, network segmentation, and intrusion detection systems, along with regular security audits and updates, help maintain the integrity and security of the WebSocket server environment.

In summary, SignalR WebSocket communication between the Backend Coordinator and the Dashboard UI incorporates multiple layers of security measures to ensure the confidentiality, integrity, and availability of real-time data. By leveraging encryption, authentication, access control, and other security features, SignalR enables the development of robust and secure real-time web applications that adhere to the highest security standards and compliance requirements.

4.3 MQTT Integration

Within the Dashboard UI, an MQTT architecture is utilized with a central MQTT broker facilitating a publish-subscribe mechanism for data ingestion from three distinct sources: UAV platform, satellite, and potentially legacy systems.

Publisher-Subscriber Interaction

- **Publishers**: UAV platform, satellite component, and HDE LSI act as publishers, generating messages with data for sharing.
- **Subscriber**: Backend Coordinator of the dashboard system is the sole subscriber, issuing subscription requests to the MQTT broker to receive data.

4.3.1 Integration with satellite component

The integration involves the Satellite Component, which is part of the Backend Coordinator system, interacting with the Dashboard UI to facilitate the prediction process based on user-defined areas of interest. We concluded with the integration as shown in Figure 35.

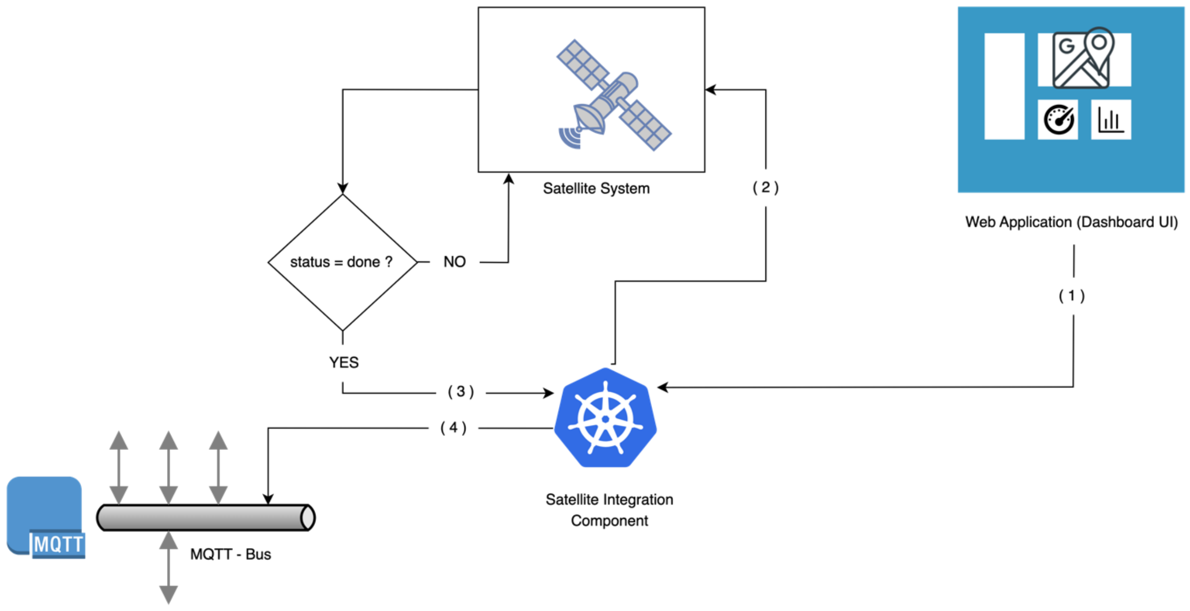


Figure 35: Satellite Component Diagram.

1. **User Interaction and Data Transmission**:
    - The end user utilizes the UI Dashboard to define areas of interest, marking them as Point or Polygon objects.
    - This information is transmitted to the Satellite Component of the Backend Coordinator for further processing.
2. **Prediction Request Initiation**:

- The Satellite component triggers a "prediction request" by sending a POST request to the /prediction endpoint.
- The request body includes a GEOJSON FeatureCollection with Point and Polygon objects for prediction.
- Upon receiving a response, the job_id is stored for future reference in retrieving results.

3. **Status Updates and Result Retrieval**:
   - The Satellite component uses the stored job_id to fetch status updates and results by sending a GET request to the /prediction/{job_id} endpoint.
   - The response includes a JSON object with fields like status (current state of the prediction job), history (progress tracking), and messages (results related to objects in the prediction request).
4. **Progress Monitoring**:
   - Continuously monitors the prediction job progress by querying the /prediction/{job_id} endpoint until the job is completed (status = done).
5. **Completion and MQTT Message Publication**:
   - Upon job completion, retrieves the "messages" section of the latest results.
   - Based on this data, the Satellite component generates and publishes an MQTT message to the MQTT broker for further processing.

This seamless integration process ensures that data from the Satellite inspection tool is effectively processed, monitored, and shared through the MQTT system for efficient communication and data dissemination within the system.

4.3.2 Integration with UAV component

The integration involves the UAV Component, a recent addition to the Dashboard's internal infrastructure, collaborating with the UAV Platform and UAV Detection System to enhance the detection process. We concluded with the integration as shown in Figure 36.

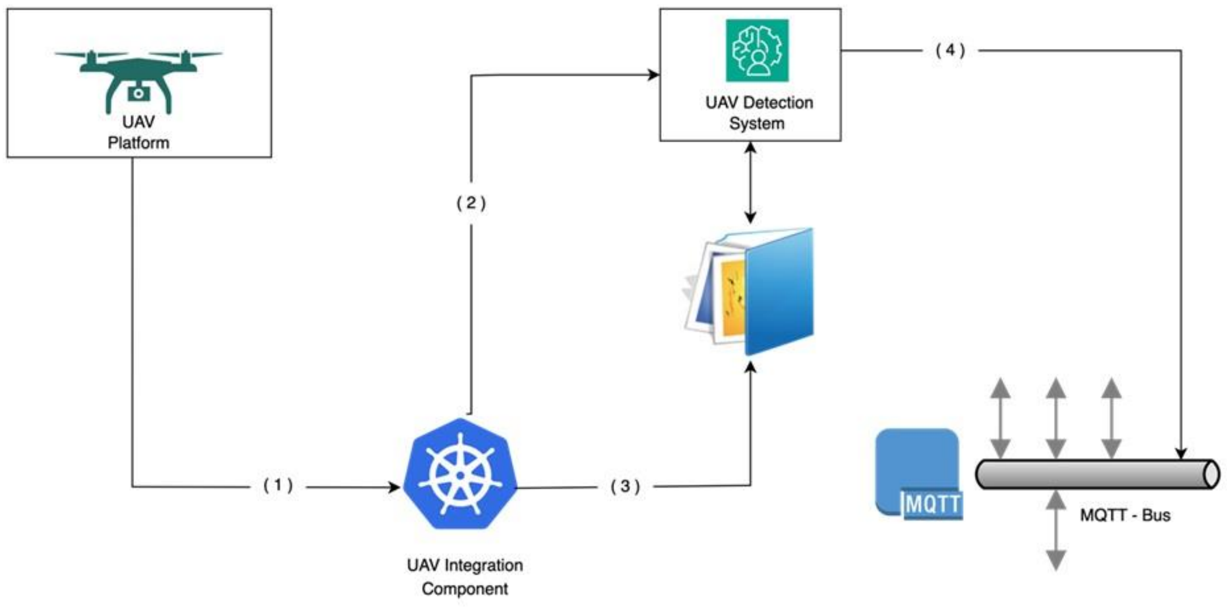


Figure 36: UAV Integration Component Diagram.

1. **Interaction with the UAV Platform:**
    - The UAV component interfaces with the UAV Platform to obtain both live video/images and stored video/images.
2. **Interaction with the UAV Detection System:**
    - The component initiates the UAV Detection System through a REST API POST call, providing settings for the UAV visual inspection routine and source details to enable real-time inspection.
    - Leveraging stored videos/images from the UAV Platform, the UAV Component sends requests and organizes files in an input folder.
    - The UAV Detection System scans all videos and images within the input folder, searching for "positive" results from detection pipelines.
    - Upon detection, the UAV Detection System generates and publishes messages containing the results of the detection algorithms on the MQTT broker.

4.3.3 Legacy systems integration

Hydro Dolomiti Energia (HDE) is the sole partner who has engaged in discussions and file exchanges to enable the running of the UI Dashboard on their legacy systems. We concluded with the integration as shown in Figure 37.

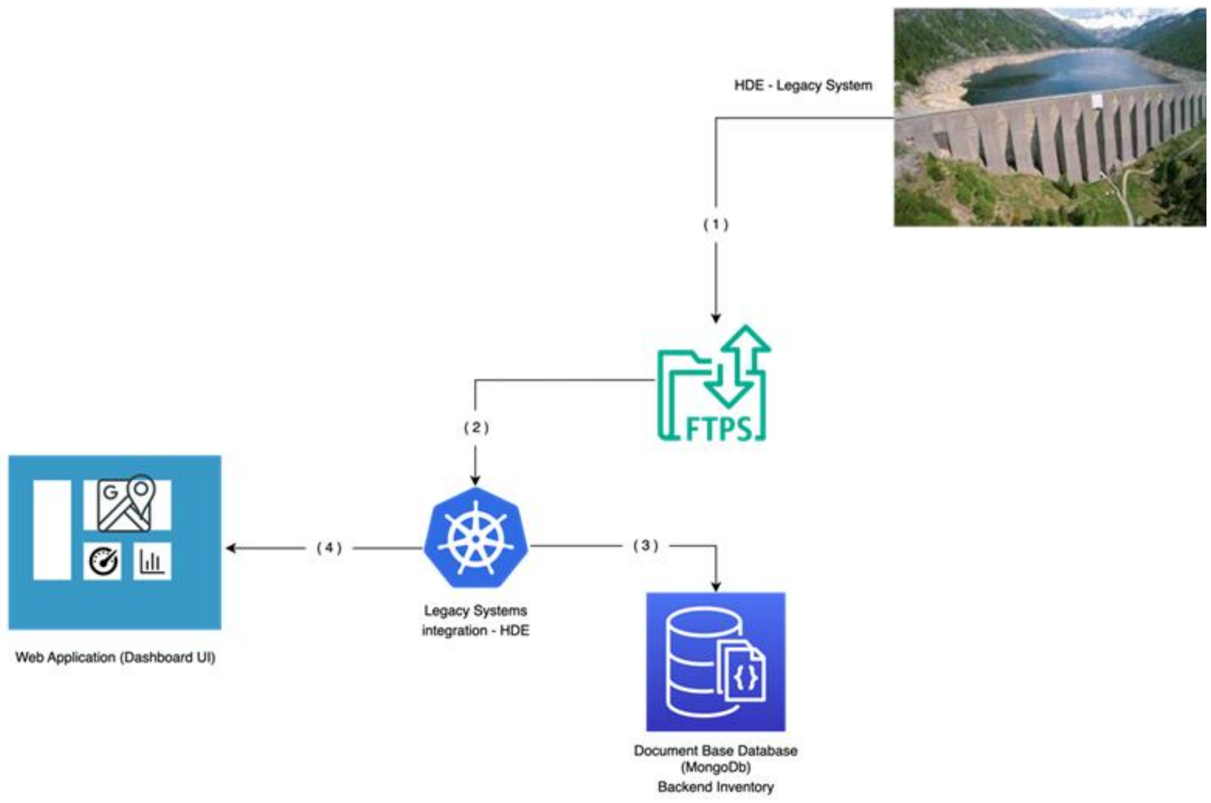

HDE - Legacy System
(1)
FTPS
(2)
(4)
(3)
Web Application (Dashboard UI)
Legacy Systems integration - HDE
Document Base Database (MongoDb) Backend Inventory

Figure 37: HDE - Legacy System Integration Diagram.

1. The HDE Legacy System, as part of its operations, creates a JSON file containing relevant data and then proceeds to transfer this file to the FTPS (File Transfer Protocol Secure) server for storage and further processing.
2. Within the system architecture, the Legacy Systems Integration (LSI) component plays a crucial role by actively monitoring a specific folder on the FTPS server. This monitoring function is designed to keep track of any new files that are deposited into this designated folder.
3. When the LSI component detects the presence of a new file in the monitored folder, it initiates a process to archive this file within the Backend Inventory system. This archival step is essential

for maintaining a comprehensive historical record of data, enabling in-depth analysis and insights over time.

4. In addition to archiving the incoming files, the LSI component is programmed to extract specific data elements known as "Properties of Interest" from the received file. These extracted properties are then transmitted to the dashboard User Interface (UI) for visualization purposes. The UI leverages this data to dynamically display the relevant information on a map, associating each property of interest with its corresponding marker for easy interpretation and analysis.

4.3.4 Legacy systems integration security

Our system implementation has been integrated with the existing legacy systems, fortified with the secure File Transfer Protocol over SSL/TLS (FTPS). Here are some reasons why FTPS is considered secure:

Encryption: FTPS utilizes encryption to safeguard data in transit. It employs SSL/TLS (Secure Sockets Layer/Transport Layer Security) protocols to encrypt the connection between the Legacy Systems client and server. This encryption ensures that data exchanged between the client and server cannot be intercepted or tampered with by unauthorized parties.

Authentication: FTPS supports various authentication methods to verify the identities of both the client (Legacy Systems) and server. These methods include username/password authentication, client certificates, and server certificates. By requiring authentication, FTPS ensures that only authorized users and servers can access the data being transferred.

Data Integrity: FTPS verifies the integrity of data during transmission to ensure that it has not been altered or corrupted. This is achieved through the use of cryptographic hash functions, which generate checksums or hash values for each data packet. The recipient can verify the integrity of the data by comparing the received hash value with the expected value.

Server Authentication: FTPS servers are typically required to present a digital certificate issued by a trusted Certificate Authority (CA). This certificate contains information about the server's identity and is used to authenticate the server to the Legacy Systems client. By verifying the server's certificate, the client can ensure that it is connecting to the correct server and not a malicious imposter.

Client Authentication: In addition to server authentication, FTPS can also require client authentication using digital certificates. This provides an additional layer of security by verifying the identity of the Legacy Systems client before allowing access to the server.

Firewall Friendly: FTPS is designed to work seamlessly with firewalls and network address translation (NAT) devices. It uses a single port (typically port 21 for control connections) for communication, making it easier to configure and manage firewall rules.

Compliance: FTPS implementations often adhere to regulatory compliance standards such as PCI DSS (Payment Card Industry Data Security Standard) and HIPAA (Health Insurance Portability and Accountability Act). Compliance with these standards ensures that sensitive data is protected during transmission, helping organizations like Legacy Systems meet their legal and regulatory requirements.

Overall, FTPS provides a secure and reliable method for Legacy Systems to transfer files over a network, making it suitable for use in environments where data security is a priority. By employing encryption, authentication, data integrity checks, and compliance with industry standards, FTPS helps Legacy Systems protect their sensitive data from unauthorized access and interception.

4.4 Deployment

The Eclipse Mosquitto MQTT broker and the MongoDB database are key components of the overall system architecture, providing the necessary messaging and data storage capabilities to support the various subsystems and the Dashboard UI.

MQTT Bus Service: this involves installing the Eclipse Mosquitto MQTT broker, version 5 or 3.1.1, on the cloud platform, configured to listen on default ports (1883). The initial setup of the broker includes creating two main topics: "UAV" for use by the UAV detection subsystem and "SAT" for use by the Satellite Component. Both subsystems publish the results of their detection processes to these topics.

Document-Based Database (MongoDB) for Backend Inventory: this entails installing the latest version of MongoDB on the cloud platform, configured to listen on default ports (27017). The initial setup of the database includes a database named "Sunrise" with two collections: "UAV" for storing all detection events from the UAV detection subsystem and "SAT" for storing all detection events from the Satellite Component. These events are later utilized for reporting and historical data visualization on the Dashboard UI.

4.4.1 Security on MQTT protocol

In our design, a comprehensive suite of security features has been integrated to enhance the integrity, confidentiality, and reliability of our MQTT communication protocol. These measures collectively fortify the security posture of our system, ensuring data protection and secure communication channels.

Transport Layer Security (TLS): the deployment of Transport Layer Security (TLS) encrypts our MQTT communication channels, safeguarding data from interception and tampering. TLS certificates, diligently managed and updated, authenticate clients and brokers, ensuring connection authenticity and enhancing defense against security breaches.

Authentication: our infrastructure enforces strict client authentication protocols through methods like username/password authentication, client certificates, and OAuth tokens. These mechanisms allow only authorized clients with valid credentials to establish connections, reducing the risk of unauthorized access to sensitive data.

Access Control Lists (ACLs): carefully configured Access Control Lists (ACLs) establish granular access control policies, governing access to specific MQTT topics based on predefined permissions. By implementing ACLs, access to sensitive data is restricted, preventing unauthorized clients from accessing beyond their designated scope.

Message Encryption: clients utilize robust application-level encryption techniques to encrypt message payloads before publishing to the MQTT broker. This additional encryption layer ensures message content confidentiality, protecting sensitive information from unauthorized access.

Broker Configuration: our MQTT broker is configured to enforce stringent security policies and mitigate common threats. Regular security audits and updates are conducted to ensure resilience against emerging vulnerabilities and threats.

Secure MQTT Implementations:

Leveraging secure MQTT broker implementations known for robust security features and proactive vulnerability management, our client libraries and SDKs are equipped with comprehensive security measures. This empowers developers to build secure MQTT applications confidently.

By implementing and maintaining these security measures, a robust and resilient MQTT infrastructure has been established, prioritizing the security and integrity of data and communication channels.

5 Pilot trials execution

The designated Pilot area should ideally be up to 2km long, facilitating the licensing process under VLOS frameworks to simplify licensing. Operating within the open category drone license framework offers flexibility and regulatory compliance for the Pilot 1 initiative.

Problematic situations should be simulated in a controlled manner to assess the algorithm's ability to identify changes. Anomalies are to be triggered securely to mimic real-world scenarios, such as a fire inside a barrel to simulate a railway fire, or a hose bleeding water to replicate a water leakage situation. The focus should be on testing the algorithm's response and accuracy in detecting these changes, rather than evaluating the UAV's endurance.

5.1 Elektro-Slovenija, d.o.o. (ELES)

ELES has dedicated significant resources and expertise in meticulously selecting the optimal locations for its pilot trials. This process involved a strategic analysis of several factors to ensure the selection of the most suitable section(s) of ELES's power lines. The criteria for this rigorous selection process included:

- Proximity to local transformer stations/substations and their control centers.
- Consideration of geographical constraints such as population density, terrain steepness, and accessibility to the infrastructure, including roads and other essential facilities.

The Infrastructure Department of ELES spearheaded the selection process, culminating in a targeted section for the pilot. Throughout the first quarter of 2024, ELES engaged in extensive collaboration with key infrastructure operators and Slovenian entities responsible for securing the necessary drone piloting licenses mandated by both local and foreign Civil Aviation Authorities (CAAs). This collaboration was pivotal due to the specialized nature of the drone licenses required, underscoring the importance of adherence to national regulations for the successful execution of the pilot trial. Moving forward, ELES will continue to provide support to Slovenian organizations and other partners in the process of obtaining necessary licenses.

The specific power line that will be used for the drone inspection is the DV 110kV Ribnica-Kočevje SM 13-31, DV Hudo-Kočevje SM 28-33; even though the entirety of power line is lengthy, a specific section will be selected where certain anomalies will be assessed. The proposed power line passes through various terrain that is adjacent (on specific sections) to forests and it also subjected to uneven terrain (hills) making it a sufficient selection to detect specific anomalies.

Since the inspection of the pylons is done manually by on-ground teams (accessed by foot) twice a year, one of the key aspects that Eles is interested in is the corrosion of the pylons and also the sag of the overhead power lines which will be beneficial for the UAV’s and it’s AI technology behind it, since it offers the possibility in reducing the human error factor in the regular inspections and additional offers the possibility of inspection without cutting the power supply. The second aspect, vegetation management, also falls within the UAV inspection since it provides continuous monitoring in aforementioned area.

5.2 Ministry of infrastructure of Slovenia (MZI)

The Ministry of infrastructure of Slovenia (MZI) has been providing assistance in procedures for obtaining authorization for the unmanned aerial vehicle (UAV) pilot trials planned in Slovenia in June 2024. The Ministry offers itself as the link between the Civil aviation agency of Slovenia and Slovene CI operators (SZ, ELES) and SKYLD who will be conducting the pilot flight trials for the visual infrastructure inspection with UAVs. The activity has been ongoing from the beginning of the year 2024.

This is important since the regulation on the unmanned aircraft systems (UAS) is relatively new and complicated therefore MZI offers it expertise to solve certain issues regarding obtaining authorizations for the upcoming pilot trials in Slovenia. These issues are also related to the national and local geographical restrictions for UAS that determine the category, operational scenarios and other requirements for the UAS operations ("Decree on implementing Regulation (EU) on the rules and procedures for the operation of unmanned aircraft" [39] and "Commission implementing regulation (EU) 2019/947 on the rules and procedures for the operation of unmanned aircraft" [40]).

5.3 Hydro Dolomiti Energia, s.r.l. (HDE)

As described in Chapter 4.1 of D7.1[41], the intake weirs of the Leno Valley have been chosen as main pilot sites. However, as the tool development and demo test design progressed, it became clear that, for the first test session, a different scenario was preferable. In fact, the inspection of the intake weirs of the Len Valley poses serious operational obstacles since it requires a BVLOS over a long distance. The requirements for such a complex flight mission are described in Annex I of D7.1[41]. Therefore, HDE analyzed again its CIs and selected the "Sarche Intake Weir". This is an intake weir on Sarca River, located roughly 5 km from Santa Massenza HPP. It is easily accessible through a paved road, and it is possible to keep VLOS all the time with the drone flying over it. The surroundings are not inhabited, and the drone will only fly over the river and the weir. Given that the latter is property of HDE, no uninvolved person is reasonably expected to be endangered by the flight of the UAV, the UAS operational scenario falls in Open sub-category A3. Therefore, no special authorization is required to fly the mission.

5.4 Designated areas

The pilots will be focusing on evaluating the performance of the Engineering inspection algorithms that have been developed thus far. The testing will be conducted in designated pilot areas with specific abnormal events, rather than focusing on drone endurance.

To facilitate the testing scenarios, it is recommended to select critical infrastructure areas that are conducive to the pilot's objectives.

It is advisable to conduct the pilot under the OPEN category instead of the SPECIFIC category, as the latter requires extensive paperwork and processes such as PDRA and SORA. This will streamline the mission and enable the pilots to focus on the critical task of testing the inspection algorithms.

All partners have agreed to operate the drone within Visual Line of Sight (VLOS) and at a safe distance from people, buildings, and recreational areas. The drone's weight is more than 4kg but under 25kg.

Adhering to these requirements, we have been classified under the Open – A3 subcategory. This subcategory is appropriate for our operation, as it allows us to maintain a clear visual line of sight and ensures the safety of people and property.

5.4.1 Piloting areas in Slovenia

The recommended area for ELES – SLOVENIA piloting is the DV 110kV Ribnica-Kočevje SM 13-31, DV Hudo-Kočevje SM 28-33 power line. The location of each sections covers mostly unpopulated areas and its also adjacent to forests.

If it is necessary to cover the entire path, it is essential to have the required equipment and a vehicle available every 1km along the route. This will ensure that the pilot aligns with the Open – A3 subcategory for drone flight and has the necessary resources to conduct the flight safely and efficiently. By taking these precautions, we can ensure a successful drone flight at the ELES facilities.

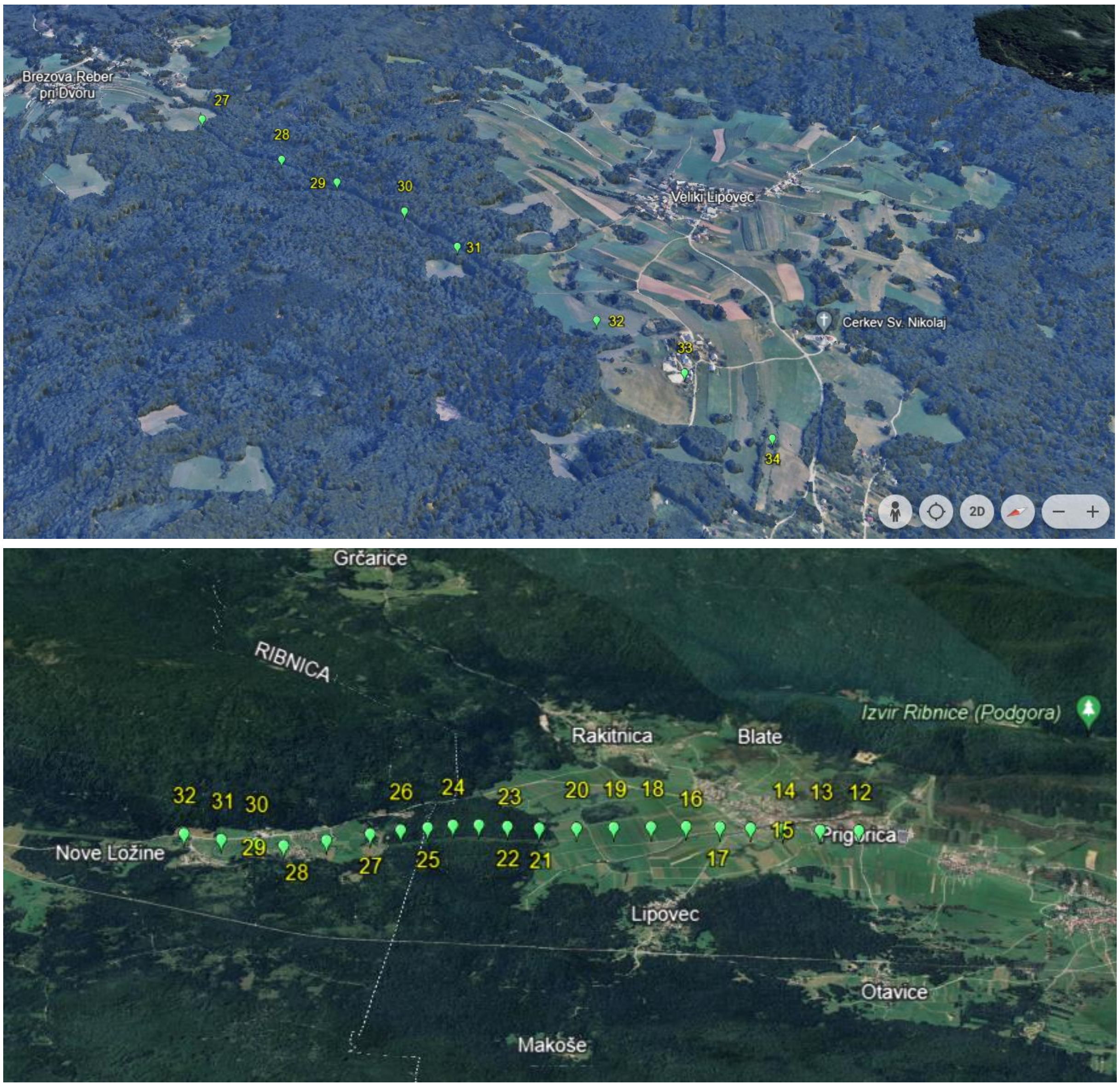


Figure 38: ELES – SLOVENIA (45°48'43"N 15°00'55"E and 45°42'52"N 14°44'59"E)

The suggested area for SZ – SLOVENIA piloting are the areas next to the train stations of Černotiči or/and Prešnica. The designated area next to the Rakek train station is also an option.

These areas offer the lowest risk environment for drone operations, being situated away from residential areas and aligns with the Open – A3 subcategory for drone flight.

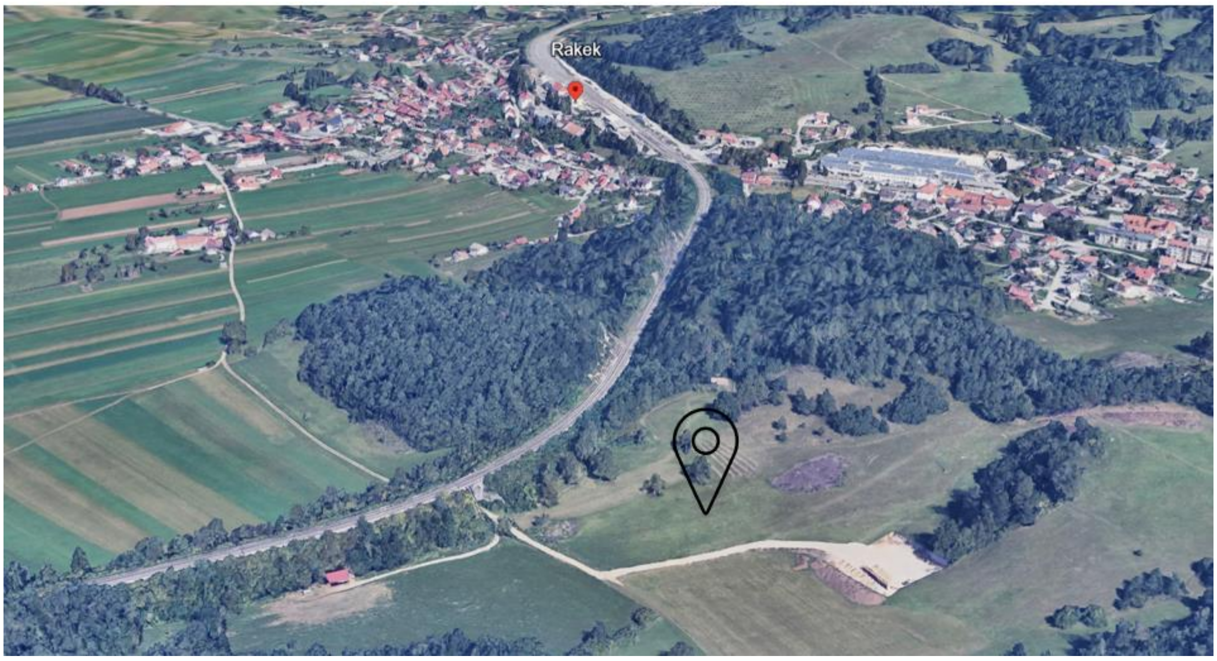


Figure 39: SZ – SLOVENIA (Rakek: 45°48'24"N 14°19'01"E).

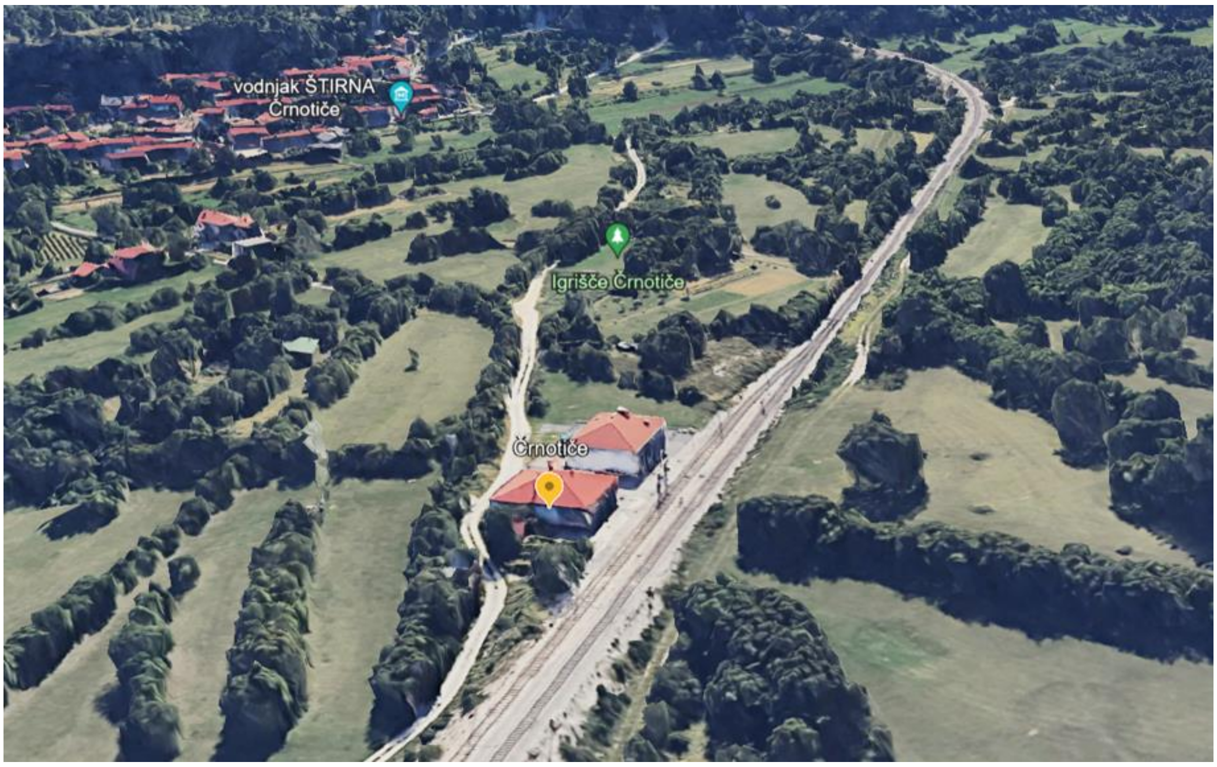


Figure 40: SZ – SLOVENIA (Černotiči: 45°33'07"N 13°53'29"E).

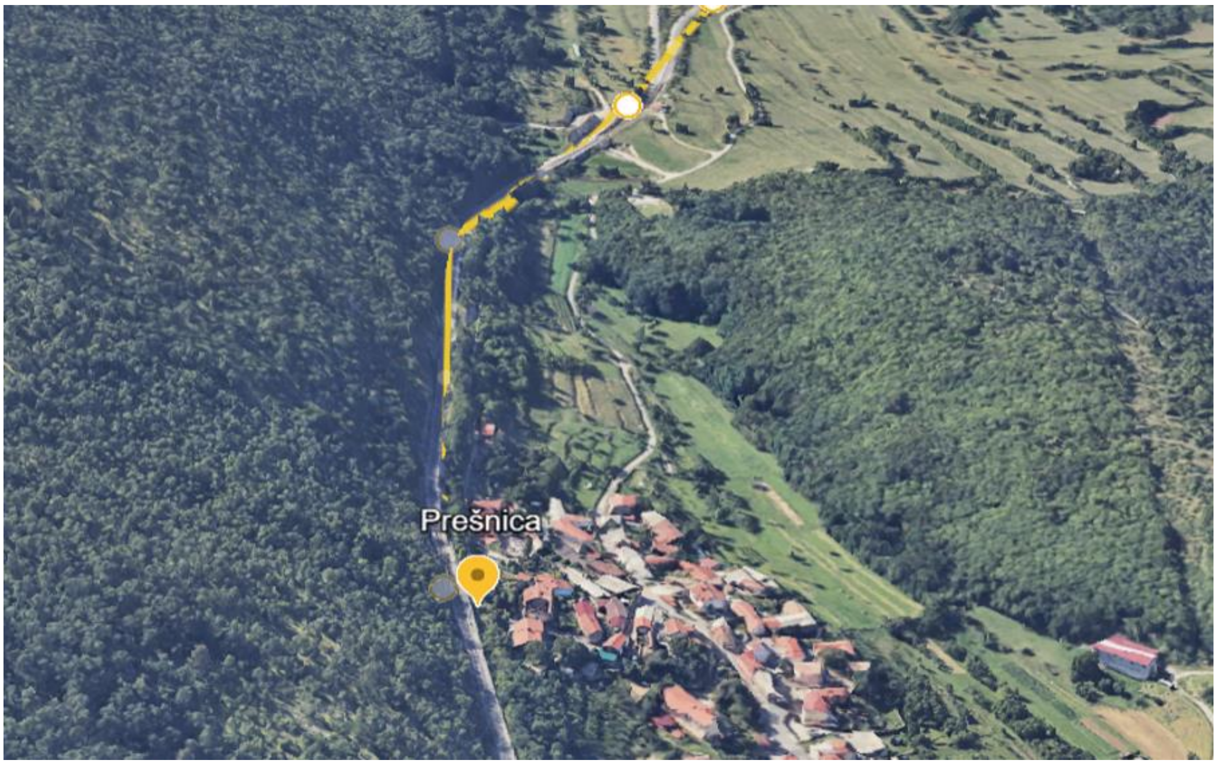


Figure 41: SZ – SLOVENIA (Prešnica: 45°34'13"N 13°56'28"E).

5.4.2 Piloting area in Italy

The proposed piloting area for HDE – ITALY is in Sarche. It is optimal case for weir inspection, where the concrete is deteriorated and there is presence of vegetation over the weir. There are also sediments in the riverbed and metal parts subjected to corrosion. Moreover, a part of the weir is covered with a wooded roof to protect it from small rocks (excellent example of 'anomalies' to be detected).

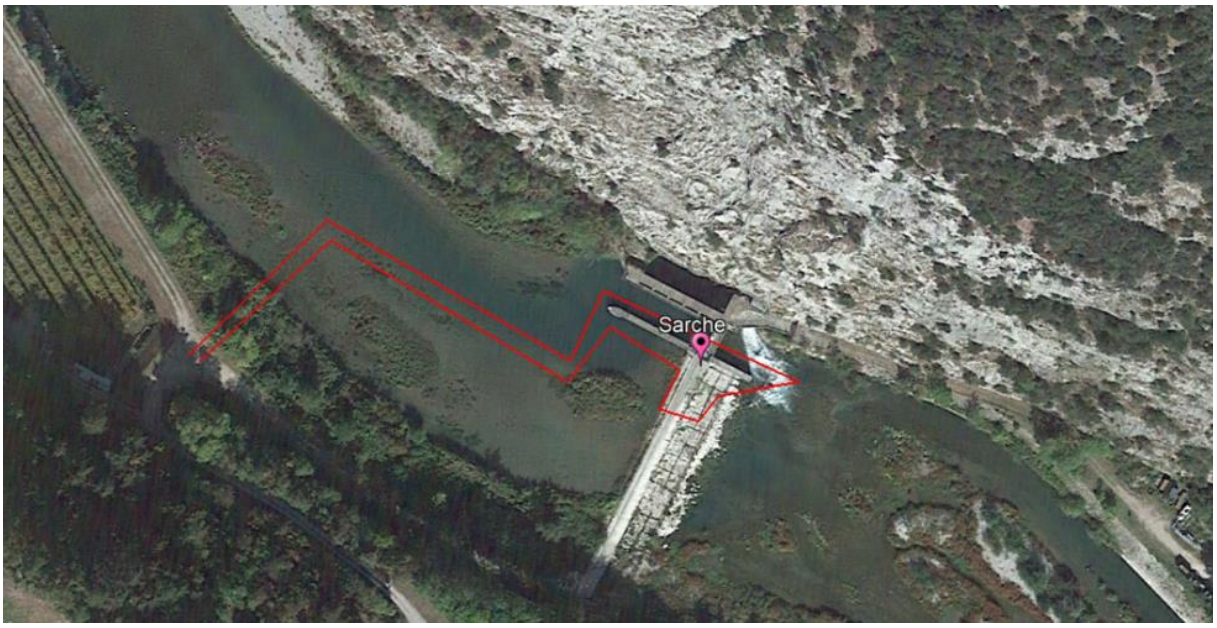


Figure 42: HDE – ITALY (46°02'50"N 10°56'44"E).

5.4.3 Piloting area in Spain

ACO has proposed an inspection area in Costa del Sol in Malaga (between Cala de Mijas and Faro de Calaburra) where both potable water and sewage pipes are in a distance of approx. 500m.

Algorithms will be tested for simulated cracks / fissures, simulated moisture / wet area, simulated damaged or manipulated manhole lids, simulated exposed pipes.

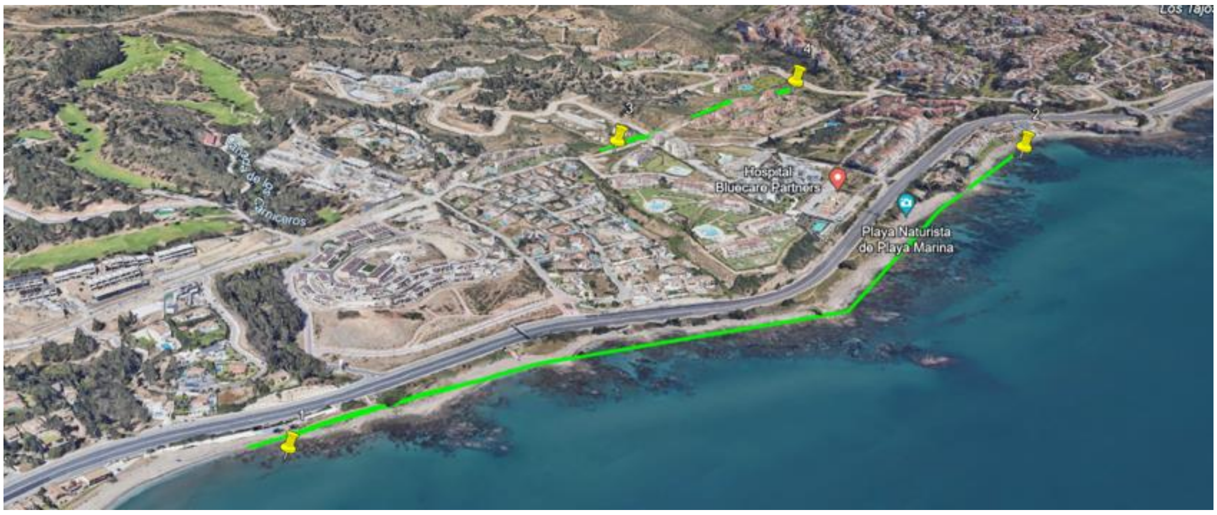


Figure 43: ACOSOL – SPAIN (36°30'23"N 4°39'02"W).

5.5 Description of End-Users' Roles

In the case of piloting the RII Tool in critical infrastructures, we summarize the personnel (their profile, role and required skills) that will be using the tool in the table below.

Table 6: Description of End-Users' Roles.

User organization	End user profile	User Role	Skills
CI operators	Strategic	Chief Technical Officer	Complete and comprehensive knowledge of the sector (policies, inspection requirements, technologies)
CI operators	Tactical	Head of Operative Unit	Technical (drone operator), data analyst
CI operators	Tactical/ Operational	Technical system specialist	Technical (drone operator), data analyst
CI operators	Operational	Data analyst	Configurator and interpreter of input data, expert knowledge on tech, drone operator
CI operators	Operational	The gathered data is used by the operational personnel. However, the data will be used also for the decisions of middle-management and in case of a big anomalies (e.g. causing bigger hindering or stop of traffic), the decisions be taken by higher management	Technical (drone operator), data analyst. Interpreter of input data (anomalies)

User organization	End user profile	User Role	Skills
CI operators	Strategic	Chief Technical Officer	Complete and comprehensive knowledge of the sector (policies, inspection requirements, technologies)
CI operators	Tactical	Head of Operative Unit	Technical (drone operator), data analyst
CI operators	Tactical/ Operational	Technical system specialist	Technical (drone operator), data analyst
CI operators	Operational	Data analyst	Configurator and interpreter of input data, expert knowledge on tech, drone operator
CI operators	Operational	Each plant manager (water treatment, desalination and sewage treatment) and the water distribution supply manager are the responsible for its implementation. General inspection of water distribution, treatment plants and pumpstations Implemented by using drones (UAV) for remote control not being subject directly to access control and also remote-controlled sensors and cameras, actually under implementations in all installations.	Technical (drone operator), data analyst. Interpreter of input data (anomalies). Very good knowledge of needs, material, and human resources, to have enough resources to guarantee production. Knowledge of security, IT, and health requirements.
CI operators	Operational	Operative personnel	Know-how of the maintenance and inspection tools and procedures.
CI operators	Operational	Technical support staff	Specialized technical knowledge in topography, structural engineering and other fields interested by inspections

6 Conclusions

This document demonstrates the progress developed for the critical infrastructures' remote inspection tool. The methodologies and algorithms employed in D7.3 have been carefully selected and implemented, to achieve the highest level of quality and accuracy of the results. The outcome will contribute significantly to the advancement for the following D7.4 and further WP7 deliverables.

The satellite image analysis compares two machine learning models for detecting changes in images. The goal is to find the best machine learning model for the specific task by comparing their performance. This involves evaluating different models based on metrics such as accuracy, bias, error distribution, and goodness of fit. The models are trained on data with different input variables and then their predictions are compared to actual values. The model with the best performance is chosen based on how well it predicts the outcome and how well it explains the variation in the output variable.

Significant advancements have been made in the UAV inspection tool, especially within the Visual Question Answering (VQA) module through the inclusion of state-of-the-art Vision-Language Models (VLLM). The addition of a face anonymization module enhances compliance with GDPR regulations, while the implementation of a new real-time open-vocabulary model expands the tool's inspection capabilities for applications requiring speed. The integration of a virtual assistant trained to simplify API interactions represents a significant step forward in user-friendliness.

These enhancements have substantially boosted the tool's functionality, positioning it as a comprehensive solution for zero-shot inspection and targeted model training. The tool now delivers valuable results, demonstrating its effectiveness in real-world scenarios.

Furthermore, substantial progress has been made in integrating this advanced UAV inspection tool with other components of the remote inspection toolset, creating a cohesive and efficient operational framework. At this stage of the project, the UAV inspection tool is well-prepared to commence the first phase of field tests. This upcoming phase is crucial for validating and refining the tool to meet the rigorous demands of critical infrastructure inspection.

In brief, the tools and methods outlined in this document mark a notable advancement in remote inspection of critical infrastructures, successfully reaching the milestone of deploying PoCs at a TRL5 or above in this project phase. The adaptable structures, integration of cutting-edge technologies, and user-friendly interface guarantee that the SUNRISE project is strongly equipped to tackle the complexities of inspecting critical infrastructures across diverse settings.

Concerning execution of pilots, we emphasize the execution of pilots in two tranches. The first is executed in M22, while the second is executed the following year. It thus logically follows that the difficulty of the pilot's implementation should be progressive. In the first practical, in the field execution of the pilots, we focus on practical aspects (like getting the necessary permits, discovering what is possible to execute given the critical nature of the critical infrastructure operators and finally, what is actually permitted given the usually quite restrictive legislation around UAV flights and the critical infrastructure).

References

[1] **Tan, M., & Le, Q. (2019, May)**. Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning (pp. 6105-6114). PMLR.

[2] **Ronneberger, O., Fischer, P., & Brox, T. (2015)** U-net: Convolutional networks for biomedical image segmentation, Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, pp. 234-241.

[3] **Zavrtanik, V., Kristan, M., & Skočaj, D. (2021)**. Reconstruction by inpainting for visual anomaly detection. Pattern Recognition, 112, 107706.

[4] **Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017)**. Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

[5] **Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017)**. Pyramid scene parsing network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2881-2890).

[6] **Toker, A., Kondmann, L., Weber, M., Eisenberger, M., Camero, A., Hu, J., ... & Leal-Taixé, L. (2022)**, Dynamicearthnet: Daily multi-spectral satellite dataset for semantic change segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21158-21167.

[7] **Gupta, R., Goodman, B., Patel, N., Hosfelt, R., Sajeev, S., Heim, E., ... & Gaston, M. (2019)**, Creating xBD: A dataset for assessing building damage from satellite imagery, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops* (pp. 10-17).

[8] **Chen, H., & Shi, Z. (2020)**, A spatial-temporal attention-based method and a new dataset for remote sensing image change detection, *Remote Sensing*, 1662.

[9] **Fang, S., Li, K., & Li, Z. (2023)**, Changer: Feature interaction is what you need for change detection, *IEEE Transactions on Geoscience and Remote Sensing.*

[10] **Chen, H., Qi, Z., & Shi, Z. (2021)**, Remote sensing image change detection with Transformers, IEEE Transactions on Geoscience and Remote Sensing, 60, 1-14.

[11] **Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016)**. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19 (pp. 424-432). Springer International Publishing.

[12] **Garnot, V. S. F., & Landrieu, L. (2021)**, Panoptic segmentation of satellite image time series with convolutional temporal attention networks, *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4872-4881.

[13] **Noh, H., Ju, J., Seo, M., Park, J., & Choi, D. G. (2022)**, Unsupervised change detection based on image reconstruction loss, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1352-1361.

[14] **Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022)**. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10684-10695).

[15] **Xu, J., Liu, S., Vahdat, A., Byeon, W., Wang, X., & De Mello, S. (2023)**. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2955-2966).

[16] **Ginzler, Christian (2021)**, Vegetation Height Model NFI. National Forest Inventory (NFI), doi:10.16904/1000001.1.

[17] **Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2017)**, Rethinking atrous convolution for semantic image segmentation, *arXiv preprint arXiv:1706.05587*.

[18] **Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B. (2021)**, Swin transformer: Hierarchical vision transformer using shifted windows, *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012-10022.

[19] **Liu, Z., Mao, H., Wu, C. Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022)**, A convnet for the 2020s. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition,* pp. 11976-11986.

[20] **Roboflow Universe**, Crack Dataset by University, https://universe.roboflow.com/university-bswxt/crack-bphdr, retrieved 2024-03-22.

[21] **Roboflow Universe**, Crack Detection Using Instance Segmentation in YOLOv8 by Bach Khoa Ho Chi Minh University, https://universe.roboflow.com/bach-khoa-ho-chi-minh-university-fyr43/crack-detection-using-instance-segmentation-in-yolov8, retrieved 2024-03-22.

[22] **Roboflow**, Corrosion Dataset, https://app.roboflow.com/project/corrosion-jzy5h/1, retrieved 2024-03-22.

[23] **Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J. & Zhang, L., 2023**. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. arXiv:2303.05499v4 [cs.CV], [online] Available at: https://arxiv.org/pdf/2303.05499.pdf, retrieved 2024-04-02.

[24] **Ke, L., Ye, M., Danelljan, M., Liu, Y., Tang, C-K., Yu, F., Tai, Y-W., 2023**. Segment Anything in High Quality. ETH Zürich & HKUST. Available at: https://arxiv.org/pdf/2306.01567.pdf, retrieved 2024-04-02.

[25] **Ultralytics, 2023**. Ultralytics GitHub Repository. [online] Available at: https://github.com/ultralytics/ultralytics, retrieved 2024-04-02.

[26] **Cheng, T., Song, L., Ge, Y., Liu, W., Wang, X., & Shan, Y., 2023.** YOLO-World: Real-Time Open-Vocabulary Object Detection. arXiv:2401.17270 [cs.CV], [online] Available at: https://arxiv.org/abs/2401.17270, retrieved 2024-04-09.

[27] **Zhou, X., Girdhar, R., Joulin, A., Krähenbühl, P. & Misra, I., 2022.** Detecting Twenty-thousand Classes using Image-level Supervision. arXiv:2201.02605v3 [cs.CV]. Available at: https://arxiv.org/pdf/2201.02605.pdf, retrieved 2024-04-09.

[28] **Zou, X., Dou, Z-Y., Yang, J., Gan, Z., Li, L., Li, C., Dai, X., Behl, H., Wang, J., Yuan, L., Peng, N., Wang, L., Lee, Y.J. & Gao, J., 2022**. Generalized Decoding for Pixel, Image, and Language. University

of Wisconsin-Madison, UCLA & Microsoft Research at Redmond. Available at: https://arxiv.org/pdf/2212.11270.pdf, retrieved 2024-04-09.

[29] **Li, J., Li, D., Savarese, S. & Hoi, S., 2023**. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. Salesforce Research. Available at: https://arxiv.org/pdf/2301.12597.pdf, retrieved 2024-04-09.

[30] **Wu, P. & Xie, S., 2023**. V*: Guided Visual Search as a Core Mechanism in Multimodal LLMs. arXiv:2312.14135 [cs.CV], [online] Available at: https://arxiv.org/abs/2312.14135, retrieved 2024-04-02.

[31] **LLaVA Team. (2024, January 30**). LLaVA: The Next Chapter. Retrieved from https://llava-vl.github.io/blog/2024-01-30-llava-next/, retrieved 2024-04-04.

[32] **Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. (2023).** Improved Baselines with Visual Instruction Tuning. arXiv:2310.03744. Available at: https://arxiv.org/abs/2310.03744, retrieved 2024-04-11.

[33] **Zhu, X., et al., 2020.** TinaFace: Strong but Simple Baseline for Face Detection. arXiv preprint arXiv:2011.13183v3. [online] Available at: https://arxiv.org/pdf/2011.13183v3.pdf, accessed 2024-04-04.

[34] Sunrise Project. (2023, December 11). **SUNRISE Tool for Remote Infrastructure Inspection Demo**. [Video]. YouTube. https://youtu.be/2zYAEmabfKs, retrieved 2024-04-11.

[35] **Papers with Code**, 2024. Face Detection Task Overview. [online] Available at: https://paperswithcode.com/task/face-detection, retrieved 2024-04-04.

[36] **Media-Smart**, 2024. TinaFace Configuration in Vedadet Repository. [online] Available at: https://github.com/Media-Smart/vedadet/tree/main/configs/trainval/tinaface, retrieved 2024-04-04.

[37] **OpenAI (2023)**. GPT-4 Technical Report. Available at: https://arxiv.org/pdf/2303.08774.pdf, retrieved 2024-04-04.

[38] **ZeroTier, 2024**. ZeroTier: Networks. [online] Available at: https://my.zerotier.com/, retrieved 2024-04-04.

[39] Republic of Slovenia, '**Decree on implementing Regulation (EU) on the rules and procedures for the operation of unmanned aircraft**' (Official Gazette of the Republic of Slovenia, no. 195/20 and no. 31/21). [online] Available at: https://pisrs.si/pregledPredpisa?id=URED8075 , retrieved 2024-03-22.

[40] European Commission, '**COMMISSION IMPLEMENTING REGULATION (EU) 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft**'. [online] Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A02019R0947-20220404 , retrieved 2024-03-22.

[41] **SUNRISE. D7.1** Infrastructure inspection conceptualization. Dejan Štepec. 2023.

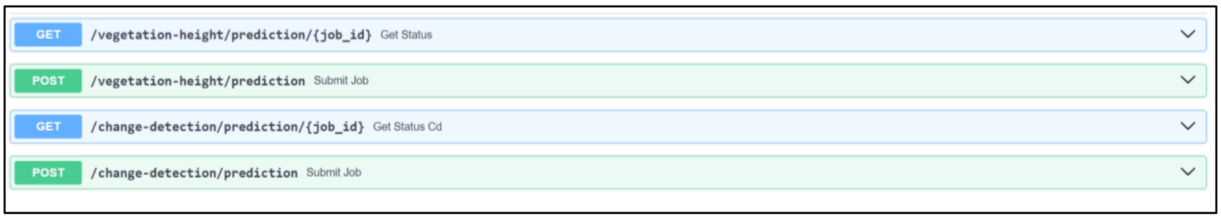
[42] **SUNRISE. D7.2** Infrastructure inspection tool and training guide V1. Mario Triviño. 2023

[43] https://www.e-plaz.si/

[44] https://mlflow.org/

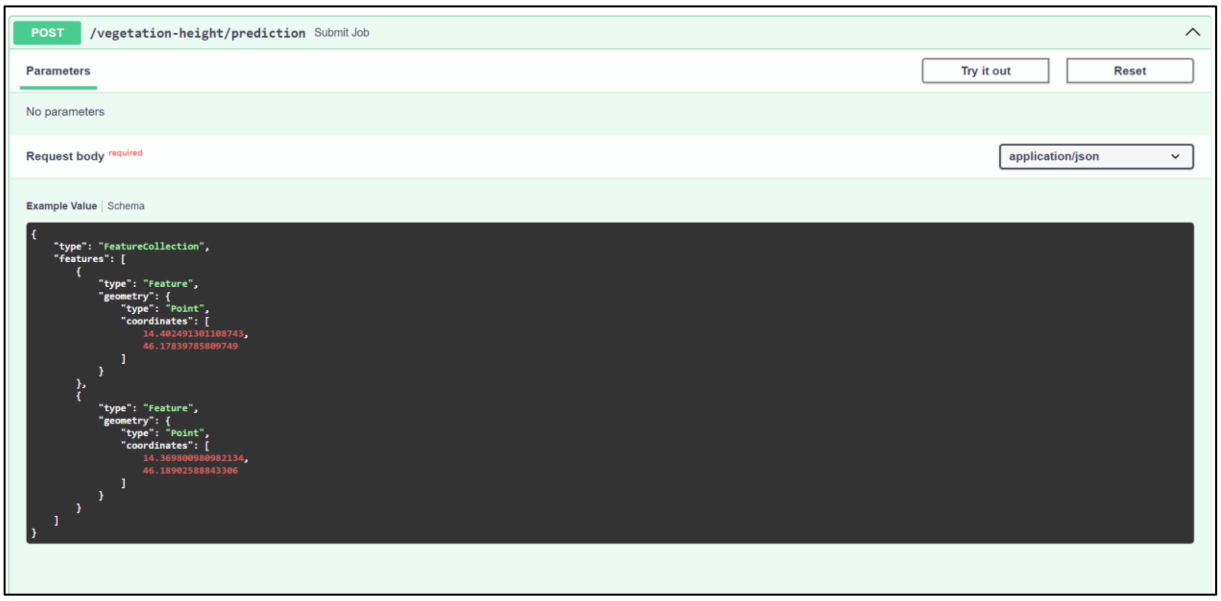
Annex I: Satellite inspection API User Guide

This section provides a user guide for the API system for the Satellite Inspection tool. The system includes two submodules, namely vegetation height and change detection, at two different endpoints. Before submitting the request, we first need to define the geographical feature describing the location where we want to perform the satellite inspection. Such features should be submitted to the API in a GeoJSON format[1]. Job can be posted to the appropriate endpoint (*/{sub-module}/prediction* where sub-module can be any of vegetation-height or change-detection) using curl command or interactively with API docs as shown in Figure 44 and Figure 45. Currently, our module supports Point and Polygon geographical features. After submitting the job, the application responds with the job_id and starts computing the output in the background. The job_id can then be used to check the status of the job and receive output.



```
GET   /vegetation-height/prediction/{job_id}   Get Status
POST  /vegetation-height/prediction            Submit Job
GET   /change-detection/prediction/{job_id}    Get Status Cd
POST  /change-detection/prediction             Submit Job
```

Figure 44: Endpoints of the satellite inspection API for posting jobs and receiving results.



POST /vegetation-height/prediction Submit Job

```
{
    "type": "FeatureCollection",
    "features": [
        {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    14.402491301108743,
                    46.17839785809749
                ]
            }
        },
        {
            "type": "Feature",
            "geometry": {
                "type": "Point",
                "coordinates": [
                    14.369800980982134,
                    46.18902588843306
                ]
            }
        }
    ]
}
```

Figure 45: Example of posting a job to vegetation-height submodule with a point geographical feature.

[1] https://geojson.org/

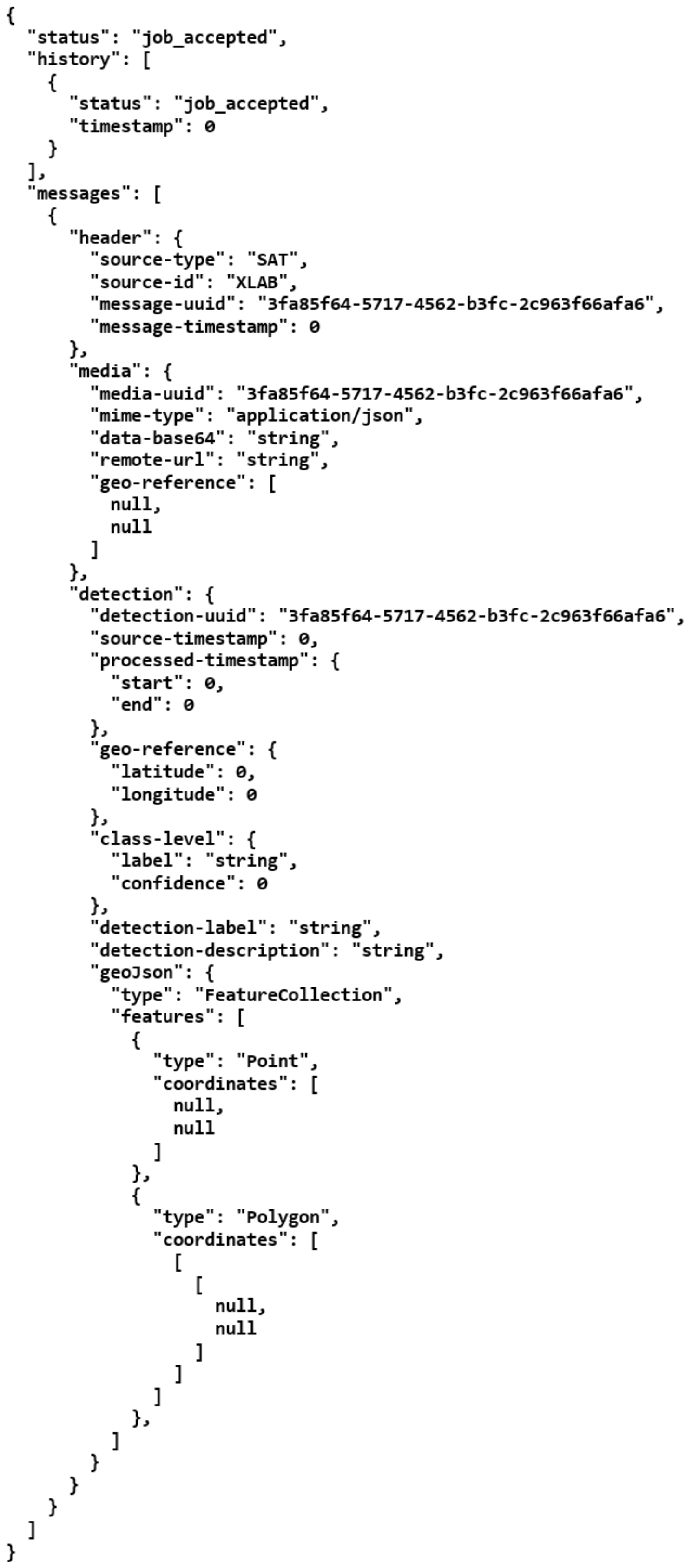
To obtain status and result, send GET request at */{sub_module}/prediction{job_id}*. Possible statuses are listed in Table 7. Full response Schema is shown in Figure 46 and is described in further detail in Table 8.

Table 7: Possible statuses of the job submitted to the satellite inspection module.

Status
Job_acepted
In_progress
Searching_satellite_images
Downloading_satelite_images
Cropping_satellite_images
Running_inference

Table 8: Description of response fields.

Field		Description
"status"		Status.
"history"		Previous statuses with timestamps.
"messages"	"header"	Basic info.
	"media"	Satellite image, encoded in base64.
	"detection"	Detection output. Result is given in GeoJSON format. Vegetation height module returns polygons with specific height of vegetation. Change detection module returns polygons where the change was detected.



```json
{
  "status": "job_accepted",
  "history": [
    {
      "status": "job_accepted",
      "timestamp": 0
    }
  ],
  "messages": [
    {
      "header": {
        "source-type": "SAT",
        "source-id": "XLAB",
        "message-uuid": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
        "message-timestamp": 0
      },
      "media": {
        "media-uuid": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
        "mime-type": "application/json",
        "data-base64": "string",
        "remote-url": "string",
        "geo-reference": [
          null,
          null
        ]
      },
      "detection": {
        "detection-uuid": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
        "source-timestamp": 0,
        "processed-timestamp": {
          "start": 0,
          "end": 0
        },
        "geo-reference": {
          "latitude": 0,
          "longitude": 0
        },
        "class-level": {
          "label": "string",
          "confidence": 0
        },
        "detection-label": "string",
        "detection-description": "string",
        "geoJson": {
          "type": "FeatureCollection",
          "features": [
            {
              "type": "Point",
              "coordinates": [
                null,
                null
              ]
            },
            {
              "type": "Polygon",
              "coordinates": [
                [
                  [
                    null,
                    null
                  ]
                ]
              ]
            },
          ]
        }
      }
    }
  ]
}
```

Figure 46: JSON response schema.

Examples of CURLs:

- Request vegetation height prediction at the given geographic coordinate:

```
curl -X 'POST' \
 'http://localhost/vegetation-height/prediction' \
 -H 'accept: application/json' \
 -H 'Content-Type: application/json' \
 -d '{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          14.402491301108743,
          46.17839785809749
  ]}},]}'
```

Response body

```
 "msg": "Prediction job accepted. Send GET to /vegetation-height/prediction to inspect the results",
 "job_id": "56cd13c4-b3b5-41d7-9aa4-4c350ad66ac6"
}
```

- Check status of the job

```
curl -X 'GET' \
 'http://localhost/vegetation-height/prediction/56cd13c4-b3b5-41d7-9aa4-4c350ad66ac6' \
 -H 'accept: application/json'
```

Response body (note that it is heavily cut).

```
{
  "status": "done",
  "history": [
   {
    "status": "job_accepted",
    "timestamp": 1714051513436
   },
      ...,
   {
    "status": "done",
    "timestamp": 1714051800326
   }
  ],
  "messages": [
   {
    "header": {
     "source-type": "SAT",
     "source-id": "XLAB",
     "message-uuid": "56cd13c4-b3b5-41d7-9aa4-4c350ad66ac6",
     "message-timestamp": 1714051800806
    },
    "media": {
     "media-uuid": "eaa45d90-0307-11ef-8025-00155dcaeeb4",
     "mime-type": "application/json",
     "data-base64": "",
     "remote-url": null,
     "geo-reference": null
    },
    "detection": {
```

```
"detection-uuid": "eaa4654c-0307-11ef-8025-00155dcaeeb4",
"source-timestamp": 1714051790806,
"processed-timestamp": {
 "start": 1714051800806,
 "end": 1714051810806
},
"geo-reference": {
 "latitude": 46.17839785809749,
 "longitude": 14.402491301108743
},
"class-level": {
 "label": null,
 "confidence": null
},
"detection-label": null,
"detection-description": null,
 "geoJson": {
   "type": "FeatureCollection",
   "features": [
     {
       "geometry": {
         "type": "Polygon",
         "coordinates": [
           [
             [
               14.406698650949064,
               46.175741985946026
             ],
               ...,
             [
               14.406698650949064,
               46.175741985946026
             ]
           ]
         ]
       },
       "property": "min=0_max=5",
       "type": "Feature"
 }]}}}]}
```

Annex II: UAV Image Processing API System User Guide

This document provides a user guide with a detailed description of the options and parameters available in the UAV image processing API system of the SUNRISE project. This system utilizes various modules to perform specific tasks on images or videos provided via Unmanned Aerial Vehicles (UAV). Table 9 lists all the input parameters that can be used, along with a brief description and an example of usage. The first decision that the user responsible for launching the inspection processes must make is which modules to use, among: 'vqa', 'segmentation', 'detection'. Once defined, the user can decide whether to use the output of one as the input to another, using "cascade" or "parallel" in the task_join parameter. Currently, the only tasks that are chained are detection or segmentation with VQA.

Once the required processes are defined, the most suitable models from those available for each task must be selected. Most parameters have a default value that allows the user to abstract away much of the API's complexity. Below are several CURL examples that help users understand the structure of the requests. In order to been able to launch the requests you will need to open a console in a system with python installed and use pip to install "requests" library.

Examples of CURLs for different inspection tasks:

- Clogged gate check:

```
curl -X POST http://api.example.com/process \
-F 'text=--tasks detection segmentation vqa \
--det_model grdSAM \
--classes_grdSAM "grate" "debris" \
--classes_grdSAM_hl "grate" \
--box_threshold_grdSAM 0.3 \
--text_threshold_grdSAM 0.3 \
--nms_threshold_grdSAM 0.5 \
--vqa_model llava \
--questions "is the grate clogged?" "is there something above the grate?" \
--seg_model_type xdecoder \
--xdec_img_size 1024 \
--vocabulary_xdec "grate" "sky" "vegetation" "soil" "people" \
--debug' \
-F 'file=@examples/clogged_grate/Clogged_grate.jpg'
```

- Fire and smoke detection:

```
curl -X POST http://api.example.com/process \
-F 'text=--tasks detection \
--det_model YOLO \
--yolo_model detection_module/models/best_fire_25000_SD.pt \
--debug' \
-F 'file=@examples/fire/fire_pexels.mp4'
```

- Custom live stream/local-cam:

```
curl -X POST http://api.example.com/process \
-F 'camera_ip=0' \
-F 'text=--tasks detection \
--det_model grdSAM \
--classes_grdSAM "people" "hammer" "glasses" "mask" \
--classes_grdSAM_hl "hammer" "sissors" \
--box_threshold_grdSAM 0.5 \
--text_threshold_grdSAM 0.5 \
--nms_threshold_grdSAM 0.5 \
--debug'
```

- Landslide detection:

```
curl -X POST http://api.example.com/process \
-F ‘text=--tasks segmentation \
--seg_model_type xdecoder \
--xdec_img_size 1024 \
--vocabulary_xdec landslide vegetation sky railway \
--debug’ \
-F ‘file=@examples/landslide/landslide_raw.jpg’
```

- Ceramic isolators inspection:

```
curl -X POST http://api.example.com/process \
-F ‘text=--tasks detection vqa \
--tasks_join cascade \
--det_model grdSAM \
--classes_grdSAM “pole” “ceramic isolators” “electrical powerline”  “catenary” “tree” “plant” “vegetation”
“cable” \
--classes_grdSAM_hl “ceramic isolators” \
--box_threshold_grdSAM 0.15 \
--text_threshold_grdSAM 0.15 \
--nms_threshold_grdSAM 0.5 \
--vqa_model seal \
--questions “describe the ceramic isolators condition status” “are the ceramic isolators in good condition” \
--vqa_triggers “broke” “rusty” \
--debug’ \
-F ‘file=@examples/isolators/1.jpg’
```

If YOLO is selected as the detection model, the specific models available are:

- Fire and Smoke: “detection_module/models/best_fire_25000_SD.pt”
- Rust/Corrosion: “detection_module/models/yolo_segx_rust.pt”
- Cracks: “detection_module/models/best_50epoch_notOT_crack.pt”

For more information or help in creating the specific curls for your task, you can use the trained virtual assistant for this task, which will also provide all the information and context you desire about the task. A subscription to ChatGPT Plus or Team is required. The link to the virtual assistant is: https://chat.openai.com/g/g-zxGORIgXT-infra-inspector-assistant.

Table 9. UAV inspection tool API's input arguments.

Module	Parameter	Description	Example Usage
Main Script	**--input**	Path to the image or video file, or stream URL.	**--input 'inputs/HDE/Example.jpg'**
	--tasks	List of tasks to be performed: 'vqa', 'segmentation', 'detection'.	**--tasks vqa segmentation**
	--tasks_join	Defines the task flow: 'cascade' or 'parallel'.	**--tasks_join cascade**
	--fps	Number of frames per second to process.	**--fps 2**
	--output	Output for visualizations.	**--output 'path/to/output/folder'**
	--debug	Indicates if debug mode is activated.	**--debug True**
	--full_pipeline	Indicates if use the full pipeline approach.	**--full_pipeline True**
	--cpu	Use CPU only for processing, not GPU.	**--cpu True**
Segmentation Module	**--seg_model_type**	Type of model for segmentation: 'xdecoder' or 'grdSAM'.	**--seg_model_type xdecoder**
	--classes_grdSAM	List of classes to detect.	**--classes_grdSAM 'class1' 'class2'**
	--classes_grdSAM_hl	List of classes to highlight with background extraction.	**--classes_grdSAM_hl 'class1'**
	--box_threshold_grdSAM	Box threshold for GroundingDINO.	**--box_threshold_grdSAM 0.3**
	--text_threshold_grdSAM	Text threshold for GroundingDINO.	**--text_threshold_grdSAM 0.3**
	--nms_threshold_grdSAM	Non-maximum suppression threshold.	**--nms_threshold_grdSAM 0.5**
	--save_imgs	Option to save resulting images.	**--save_imgs True**
	--config_file_xdec	Path(s) to the config file(s) for the X-Decoder.	**--config_file_xdec 'config_file_path.yaml'**

Module	Parameter	Description	Example Usage
	--xdec_img_size	Reshape size for the image to be processed with X-Decoder.	**--xdec_img_size 512**
	--vocabulary_xdec	Concepts for segmentation with X-Decoder.	**--vocabulary_xdec 'concept1' 'concept2'**
	--vocabulary_xdec_hl	Concepts for segmentation with highlighting.	**--vocabulary_xdec_hl 'concept1' 'concept2'**
	--xdec_pretrained_pth	Path(s) to the weight file(s) for X-Decoder.	**--xdec_pretrained_pth 'path/to/weights.pt'**
	--xdec_type	Type of segmentation with X-Decoder: 'semseg' or 'refseg'.	**--xdec_type semseg**
	--pred_all_class	Option to predict all classes.	**--pred_all_class**
Detection Module	**--det_model**	Select the model used for detection: 'Detic', 'grdSAM' or 'YOLO'.	**--det_model YOLO**
	--config-file-detic	Path to the config file for Detic.	**--config-file-detic 'path/file.yaml'**
	--vocabulary	Vocabulary used in Detic: 'lvis', 'openimages', etc.	**--vocabulary custom**
	--custom_vocabulary	Custom vocabulary for Detic.	**--custom_vocabulary 'object1,object2'**
	--confidence-threshold	Minimum threshold for displaying predictions.	**--confidence-threshold 0.2**
	--nms_max_overlap	Maximum overlap threshold in NMS.	**--nms_max_overlap 0.3**
	--patching	Option to process the image in patches.	**--patching True**
	--patch_size	Size of the patches.	**--patch_size 336**
	--overlap	Overlap of the patches.	**--overlap 0.3**
	--yolo_model	Path to the YOLO model.	**--yolo_model 'detection_module/models/model_YOLO.pth'**
VQA Module	**--vqa_model**	Select the model for visual question answering: "blip", "llava" or "seal".	**--vqa_model 'seal'**

Module	Parameter	Description	Example Usage
	--questions	List of questions to be answered by the AI.	**--questions 'What is in the image?'**
	--vqa_triggers	List of concepts to check if appears in the answers.	**--vqa_triggers 'problem'**
	--conv_type	Type of conversation you want.	**--conv_type 'custom_infra'**