

ASSISTANCE

Adapted situation awareneSS tools and tallored training curricula for increaSingcapabiliTie and enhANCing the proteCtion of first respondErs



European Commission

Project co-funded by the European Union within the Horizon 2020 Programme



Project Ref. N°	ASSISTANCE H2020 - 832576
Start Date / Duration	May 1, 2019 (36 months)
Dissemination Level ¹	PU (Public)
Author / Organisation	Łukasiewicz - PIAP

Deliverable D4.3

Robots integrated into the system

31/07/2021

¹ PU: Public; PP: Restricted to other programme participants (including the EC services); RE: Restricted to a group specified by the Consortium (including the EC services); CO: Confidential, only for members of the Consortium (including the EC services).

ASSISTANCE

Nowadays different first responder (FR) organizations cooperate together to face large and complex disasters that in some cases can be amplified due to new threats such as climate change in case of natural disasters (e.g. larger and more frequent floods and wild fires, etc) or the increase of radicalization in case of man-made disasters (e.g. arsonists that burn European forests, terrorist attacks coordinated across multiple European cities).

The impact of large disasters like these could have disastrous consequences for the European Member States and affect social well-being on a global level. Each type of FR organization (e.g. medical emergency services, fire and rescue services, law enforcement teams, civil protection professionals, etc.) that mitigate these kinds of events are exposed to unexpected dangers and new threats that can severely affect their personal safety.

ASSISTANCE proposes a holistic solution that will adapt a well-tested situation awareness (SA) application as the core of a wider SA platform. The new ASSISTANCE platform is capable of offering different configuration modes for providing the tailored information needed by each FR organization while they work together to mitigate the disaster (e.g. real time video and resources location for firefighters, evacuation route status for emergency health services and so on).

With this solution ASSISTANCE will enhance the SA of the responding organisations during their mitigation activities through the integration of new paradigms, tools and technologies (e.g. drones/robots equipped with a range of sensors, robust communications capabilities, etc.) with the main objective of increasing both their protection and their efficiency.

ASSISTANCE will also improve the skills and capabilities of the FRs through the establishment of a European advanced training network that will provide tailored training based on new learning approaches (e.g. virtual, mixed and/or augmented reality) adapted to each type of FR organizational need and the possibility of sharing virtual training environments, exchanging experiences and actuation procedures.

ASSISTANCE is funded by the Horizon 2020 Programme of the European Commission, in the topic of Critical Infrastructure Protection, grant agreement 832576.

D4.3 Robots integrated into the system

Disclaimer

This document contains material, which is the copyright of certain ASSISTANCE consortium parties, and may not be reproduced or copied without permission.

The information contained in this document is the proprietary confidential information of the ASSISTANCE consortium (including the Commission Services) and may not be disclosed except in accordance with the consortium agreement.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the project consortium as a whole nor a certain party of the consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, and accepts no liability for loss or damage suffered by any person using this information.

The information in this document is subject to change without notice.

Executive Summary

This deliverable describes the integration of the UGV and its subsystems with the ASSISTANCE Situational Awareness Platform (SAP) through the Sensor Abstraction Service (SAS), based on the ASSISTANCE system architecture described in deliverable D2.4 ASSISTANCE System and Network Architecture Design.

It reports on the results of task T4.3: Robots Management and sensors integration and summarises modifications and new developments related to the UGV platform and all its components as they have been integrated within the ASSISTANCE system and in relation to the respective robot requirements and use-cases as described in deliverables D2.2 and D2.4.

List of Authors

Organisation	Authors
Łukasiewicz-PIAP	Tomasz Plaskota, Michał Chomej, Michał Bryła, Mateusz Maciaś, Agnieszka Sprońska, Łukasiewicz-PIAP Team

Change control datasheet

Version	Changes	Chapters	Pages	Date
0.1	First draft	1	11	17/05/2021
0.2	ToC prepared	All	16	25/05/2021
0.5	Updates to chapters 1,2,3,4,6	1,2,3,4,6	37	22/06/2021
0.6	Updates to chapters 4,5,6,7	4,5,6,7	91	12/07/2021
0.9	Updates after internal Łukasiewicz-PIAP review	All	66	20/07/2021
1.0	Updates after ASSISTANCE partners review: UPV and OSPOM	All	66	30/07/2021

Table of Content

Executive Summary	3
List of Authors	4
Change control datasheet	5
Table of Content	6
List of Figures	9
List of Tables	10
Acronyms	11
1. Introduction.....	13
1.1. Purpose of the document	13
1.2. Scope.....	13
1.3. Relationship with other work packages	14
2. ASSISTANCE General Architecture	15
3. Pilots Scenarios.....	17
3.1. Scenario 1.....	17
3.2. Scenario 2.....	17
3.3. Scenario 3.....	18
4. Robot platform	19
4.1. Robot hardware overview	19
4.1.1. PIAP GRYF® platform	19
4.1.2. Autonomy module.....	20
4.1.3. Enclosure	21
4.1.4. Communication	22
4.2. Robot software overview.....	22
4.2.1. Low-level computer	22
4.2.2. Video server.....	23
4.2.3. ROS architecture.....	23
4.2.4. JAUS architecture	24
4.2.5. Project specific optimizations.....	24

D4.3 Robots integrated into the system

5.	GCS.....	25
5.1.	GCS architecture	25
5.1.1.	General architecture.....	25
5.1.2.	Web User Interface functional modifications	26
5.2.	SAS Adapter	27
5.2.1.	SAS Adapter interfaces.....	27
5.2.2.	MQTT interface	28
5.2.2.1.	Resource	28
5.2.2.2.	Sensor	29
5.2.3.	NATS adapter	30
5.2.3.1.	Context	31
5.2.3.2.	Mission.....	31
5.2.4.	Sensor action	32
5.3.	J AUS compatibility	33
5.3.1.	Implemented services	33
5.4.	Data streams	35
5.4.1.	JANUS gateway	35
5.4.2.	Sensor data	35
5.5.	User interface	37
5.5.1.	Requirements	37
5.5.2.	General view	38
5.5.3.	Waypoint editor.....	40
5.5.4.	Layer editor.....	40
5.5.5.	Services.....	41
5.5.6.	Context management.....	42
6.	Payloads.....	44
6.1.	Common elements.....	44
6.1.1.	Common electronic interface.....	45
6.1.2.	Common communication interface.....	45
6.1.3.	Common physical interface	45
6.2.	RGB Cameras.....	45

D4.3 Robots integrated into the system

6.3.	Thermal camera	46
6.3.1.	Initial research	46
6.3.2.	Hardware	47
6.3.3.	Video transcoding.....	49
6.3.4.	Testing	49
6.4.	ATMON FL gas analyser	50
6.4.1.	Measured substances.....	50
6.4.2.	Driver development.....	51
6.4.3.	Sensor simulator	52
6.5.	MG-811CO2 sensor	52
6.5.1.	Sensor calibration	54
6.5.2.	Testing	55
6.6.	ZR-2 radioactivity detector	56
6.6.1.	Sensor description	56
6.6.2.	Modifications.....	56
6.7.	VAISALA WXT-520 weather station	57
6.7.1.	Sensor description	57
6.7.2.	Modifications.....	57
6.8.	EMF detector	57
6.8.1.	Market analysis.....	58
6.8.2.	Hardware	59
6.8.3.	Firmware.....	61
6.8.4.	Calibration	61
6.9.	CUTLANCA cutting extinguisher.....	63
6.9.1.	Concept.....	63
6.9.2.	Mechanical integration.....	63
6.9.3.	Testing	64
7.	Summary.....	64

List of Figures

Figure 1 ASSISTANCE Architecture Schema.....	15
Figure 2 ASSISTANCE Network Architecture schema	17
Figure 3 Robot platform general overview	19
Figure 4 PIAP GRYF®	20
Figure 5 PIAP- GRYF enclosure	21
Figure 6 UGV communication architecture.....	22
Figure 7 Dependencies between nodes in the autonomy module (left) and exemplary ROS-JAUS-Bridge ROS node (right).....	23
Figure 8 GCS components	25
Figure 9 Janus Gateway	35
Figure 10 Sensor data exchange information	36
Figure 11 GCS main view	38
Figure 12 Access Control	39
Figure 13 Editing waypoints on map	40
Figure 14 Layer editor.....	41
Figure 15 All services view with default settings	41
Figure 16 Example service card	42
Figure 17 Context assignment component	43
Figure 18 Quick release NATO rail clamps and recess for gripper	45
Figure 19 Thermal imaging camera, mounted on the UGV	46
Figure 20 Thermal camera electronics architecture	48
Figure 21 Finished thermal camera electronics and custom made enclosure	49
Figure 22 Example photos taken with thermal camera	49
Figure 23 ATMON FL main unit	50
Figure 24 Atmon related ROS message schemas	52
Figure 25 MG-811 sensors, integrated with the UGV	53
Figure 26 Base plate with mounted electronics and assembled device	54
Figure 27 Calibration setup for multiple physical sensors	54
Figure 28 Data collected during the calibration process.....	55
Figure 29 ZR-2 radioactivity sensor	56
Figure 30 Vaisala WXT520 weather station	57
Figure 31 EMF sensor mounted in the UGV gripper	58
Figure 32 AC HotStick	58
Figure 33 RS Pro Multi-field EMF meter.....	59
Figure 34 - EMF measurement circuit architecture	60
Figure 35 - EMF sensor electronics.....	60
Figure 36 3D-printed EMF sensor enclosure	61

D4.3 Robots integrated into the system

Figure 37 EMF output on different ranges after scaling	62
Figure 38 CUTLANCA mechanical integration result	63

List of Tables

Table 1 Intel® NUC specifications	21
Table 2 Resource message	29
Table 3 Sensor message	30
Table 4 Context message	31
Table 5 Mission message	32
Table 6 GCS main view icons descriptions	40
Table 7 ATMON FL measured substances	51

Acronyms

AB	Advisory Board
ADC	Analog-Digital Converter
API	Application Programming Interface
ASSISTANCE	Adapted situation awareneSS tools and tallored training curricula for increaSingcapabiliTie and enhANCing the proteCtion of first respondErs
CAN	Controller Area Network
COTS	Commercial off-the-shelf
CPU	Central Processing Unit
D#.#	Deliverable number #.# (D1.1 deliverable 1 of work package 1)
DALR	Damaged Assets Location and Routing
DCMI	Digital Camera Interface
DoA	Description of Action of the project
EC	European Commission
EOD	Explosive ordinance disposal
EU	European Union
FOV	Field of View
FPGA	Field-programmable gate array
FR	First Responder
GA	Grant Agreement
GCS	Ground Control Station
GNSS	Global Navigation Satellite Systems
GPIO	General purpose input/output
GPS	Global Positioning System
GPU	Graphics processing unit
GUI	Graphical User Interface
H2020	Horizon 2020 Programme for Research and Innovation
I2C	Inter-Integrated Circuit
IMU	Inertial measurement unit
IPR	Intellectual Property Rights
IR	Infrared
J AUS	Joint Architecture for Unmanned Systems
LIDAR	Light Detection and Ranging
M#	#th month of the project (M1=May 2019)
MQTT	Message Queuing Telemetry Transport
NATS	Neural Autonomic Transport System

D4.3 Robots integrated into the system

NUC	Next Unit Computing
PAL	Phase Alternating Line
PC	Personal Computer
PCB	Printed Circuit Board
PET-G	Polyethylene terephthalate glycol
PIC	Project Implementation Committee
PSB	Project Security Board
PSC	Project Steering Committee
PTZ	Pan Tilt Zoom
RDB	Reference Design Board
REST	REpresentational State Transfer
RGB	Red Green Blue
ROS	Robot Operating System
ROS-JAUS-Bridge	Software bridge between ROS and JAUS transfer protocols
RTP	Real-time Transport Protocol
RTSP	Real Time Streaming Protocol
SAP	Situational Awareness Platform
SAS	Sensor Abstraction Service
SOC	System on Chip
SoC	System on Chip
SPI	Serial Peripheral Interface
TL	Task Leader
UART	Universal asynchronous receiver-transmitter
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
UGV	Unmanned Ground Vehicle
UWB	Ultra WideBand
UxV	Any Unmanned Vehicle – UAV/UGV
WP	Work Package
WPL	Work Package Leader

1. Introduction

This document reports on the work performed in task T4.3 Robots Management and sensors integration that was required to achieve full integration with the ASSISTANCE system. This includes modifications to base robot platform, GCS as well as development and integration of required sensors. All the developments presented in this deliverable have been planned according to requirements and scenario's and use-cases specified in deliverables D2.2 and D2.3 and analysed within task T4.1 to be outlined in deliverable D4.1, which describes the UGV platform selection and lists in detail all its required adaptations. Integration of the robot platform into the ASSISTANCE system has been performed following the system architecture designed in deliverable D2.4 and using the interfaces defined in deliverable D3.1.

1.1. Purpose of the document

The main purpose of this document is to describe the integration of the UGV platform and its components highlighting the most important parts of a multi-level architecture of the robot system as relevant to the ASSISTANCE system.

In the project context this document is a continuation of deliverable D4.1 and constitutes a final document related to the UGV platform adaptation and integration. Its release finalises the achievement of the project milestone MS4 "Unmanned platforms and control devices integrated".

1.2. Scope

This deliverable encompasses the description of the adapted UGV platform and the integration of its payloads with the rest of ASSISTANCE system, which is one of the key results of WP4.

First, in Chapter 2 the necessary context within the project is provided in a form of the ASSISTANCE general architecture and description of the correlation between the UGV platform and this architecture. Next, in Chapter 3 the connections between described work performed and the 3 demonstration scenarios are established.

Following the general chapters 3 major technical parts are included:

- Robot platform: description of the selected UGV with relevant adaptations
- GCS: description of the UGV control software with required modifications
- Payloads: description of the developed/adapted sensors/actuators

1.3. Relationship with other work packages

This task takes and builds up on the results from the following tasks:

- T2.2 User requirements gathering analysis and tracking.
- T2.3 Reference Scenarios, Pilot Operations Specifications and KPIs.
- T2.4 System and Network Architecture Design.
- T3.1 Sensor Abstraction Service Adapted Interfaces Definition.
- T3.2 Sensor Abstraction Service Adapted Interfaces Implementation.
- T4.1 Unmanned Platforms Selection & Adaptation.

This task contributes to the following tasks:

- T4.6 Mission management.
- T5.4 Advanced Modules, SAS & Communications Infrastructure Integration in ASSISTANCE SA Platform.
- T7.1 Validation Plan.
- T7.2 Integrated system test bed.
- T7.3 Pilot Demonstration.

2. ASSISTANCE General Architecture

Deliverable D2.4 describes in detail the ASSISTANCE architecture and is therefore a basis for understanding the UGV position within the system. In this section a summary of the most relevant elements from ASSISTANCE general architecture are provided in the context of robots.

The high level architecture schema (Figure 1) as taken from Chapter 3 of deliverable D2.4 shows that UxVs communicate with other independent system components via interfaces provided in the Sensor Abstraction Service (SAS).

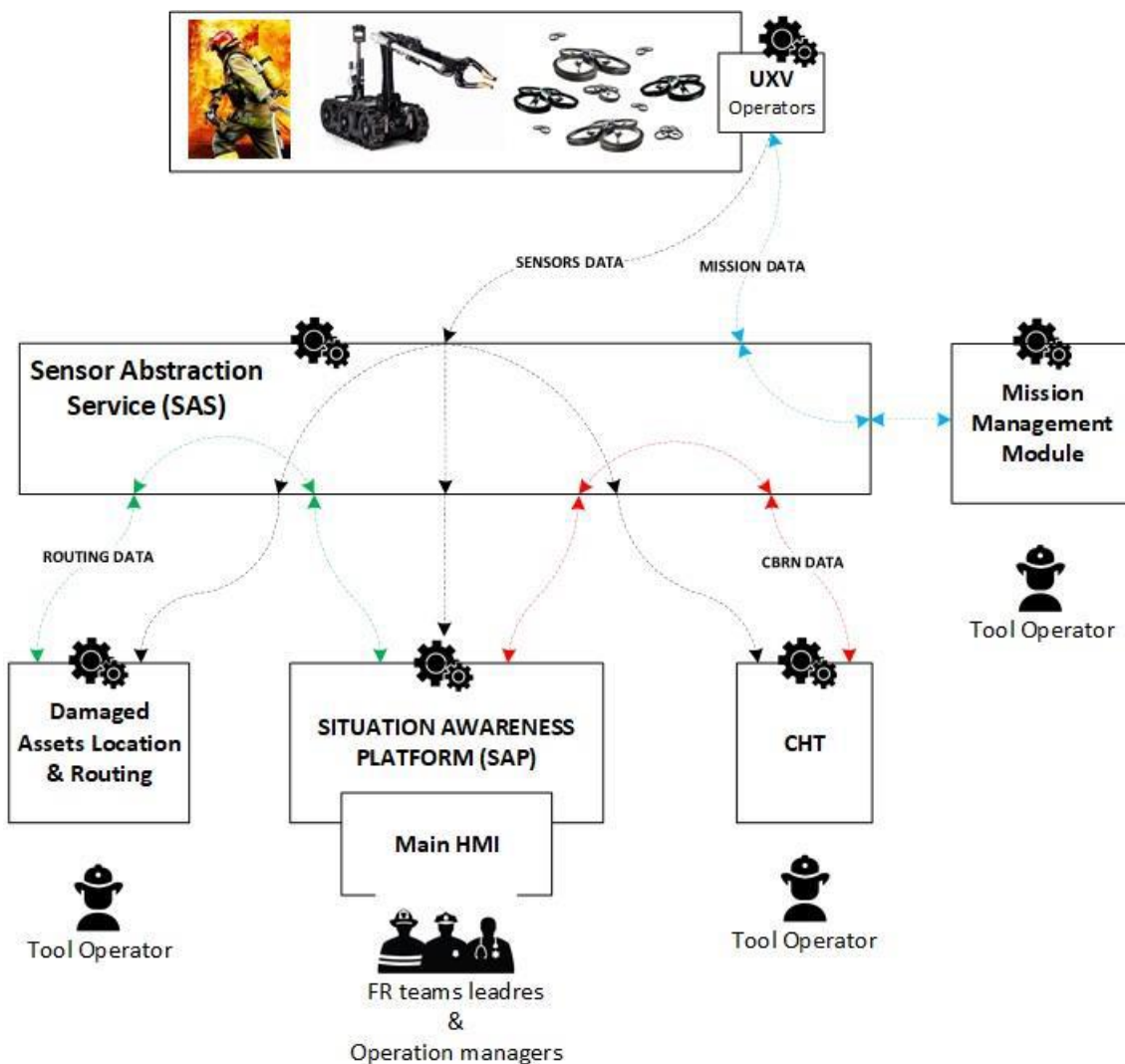


Figure 1 ASSISTANCE Architecture Schema

As presented in Figure 1, the UxVs have two types of data transfer channels available – unilateral sensor data channel from an UxV to SAS and bilateral mission data channel. Sensor data is separated in 2 types of data:

D4.3 Robots integrated into the system

- Video stream – RTSP H.264 video,
- Raw sensor data – raw data captured by sensor payloads running on robot;

while mission data provides other data such as:

- Reporting robot complete information including position, attitude, altitude, velocity, identification data, current context assignment, battery life and complete lists of attached cameras with their positions as well as lists of currently attached sensors,
- Defining characteristics of sensors and data accuracy,
- Replicating list of available contexts for data reporting,
- Receiving mission parameters for execution including list of waypoints, list of actions to perform during mission and context to report data during mission execution.

Serialization format for both communication channels is defined in the ASSISTANCE data model that has been developed in WP3. The data model is much broader than only communication with the UxVs – it also provides additional communication to all independent modules developed in the project. The design principle behind the SAS middleware is to allow indirect communication of the components without exposing end-points to each component. This is achieved by all communication logic and dynamic component management to be performed by the SAS. In this way no additional component, except middleware, is required to have prior knowledge about rest of components deployed within scope of ASSISTANCE system.

Additionally, the SAS implements multiple protocols to facilitate independence of each module even further – in this setup to have a working communication it is only required to follow the common data model. Each module can implement a method of communication that is best suited for the type of application. The available protocols are: MQTT, DDP/NATS and REST API.

D4.3 Robots integrated into the system

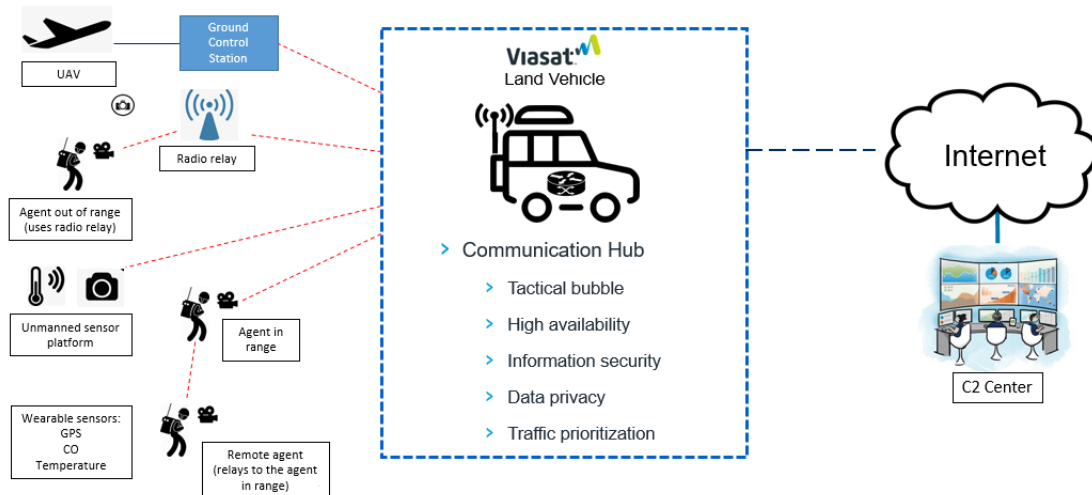


Figure 2 ASSISTANCE Network Architecture schema

ASSISTANCE also provides a network solution (Figure 2) that will be used during the project demonstrations. For UxVs this results in no additional requirements for connection with SAP, except having to have the GCS in the range of the Communication Hub vehicle. As shown in Figure 2, all data produced by the robots is being first sent to the GCS which is responsible for communication with the ASSISTANCE middleware.

3. Pilots Scenarios

In this chapter a brief overview of each scenario is described in the context of the use of the UGV. The complete pilots scenarios descriptions are published in deliverable D2.3 ASSISTANCE Reference Scenarios and Pilot Experiment.

3.1. Scenario 1

Scenario 1 will be performed in Izmir, Turkey and it's going to showcase the capabilities of the ASSISTANCE system in an earthquake disaster scenario. In this scenario the robot was required to provide feed from cameras to C2 and DALR tool.

During the earlier stages of the project and specifically during the detailed planning in task T4.1 a risk with logistical and legal problems of transferring the selected robot to Turkey were identified. This risk was analysed and discussed in the project consortium and decision taken was to not to include ground robots in the first project scenario.

3.2. Scenario 2

The second pilot will take place in Rotterdam (the Netherlands) and its main focus will be fire disaster in a heavily industrialized area with potential leakage of toxic substances.

D4.3 Robots integrated into the system

The ground robot will be used in several use-cases in this scenario. UC1 involves the UGV deployment, UC2 and UC3 extend the deployment with a whole system connectivity verification. UC4 and UC5 are related to the SAP and DALR presentation, during which the UGV will provide sensor data and video streams for display. UC6 is a coordinated investigation mission with FRs and the UGV inside a building. Finally, in UC8 the UGV will be used to provide sensor readings for the CBRN module.

During all uses cases of pilot 2 a subset of available sensors will be used depending on the specific use case needs. In UC1 all the sensors will be mounted and checked for proper communication, in UC6 the EMF, ATMON FL, MG811 and thermal camera will be required and in UC8 the weather station, ATMON FL, MG811, ZR2 will be deployed.

Also, at the request of the end-users additional presentation will be held involving disarming acetylene container using the cutting extinguisher.

3.3. Scenario 3

Third pilot will take place in Villacarrillo, Spain in ATLAS facility premises and will be based on simulation of a terrorist attack.

In this scenario two ground robots will be used. PIAP-GRYF that is being described in this document as well as ANUAV Robot provided by MIR-PN. ANUAV Robot will be equipped with multiple sensors and will be connected with ASSISTANCE system only with a telemetry module providing positional information.

Similarly to Scenario 2, the UGV (PIAP-GRYF) will be used in multiple use cases. UC1 will involve the UGV deployment, UC2 and UC4 will extend the deployment with a whole system connectivity verification. UC5 and UC8 will be the SAP and DALR presentation, during which the UGV will provide sensor data and video streams for display.

During the demonstrations both robots will be used for joint coordination exercises.

4. Robot platform

In this chapter the components of the robot platform are described. As shown in Figure 3 in this deliverable the robot platform will be referred to as the UGV with an autonomy module and low-level controller as well as payloads mounted. Payloads are therefore considered a part of the robot platform, but since they are modular and quite a broad topic in this deliverable they are described in detail in Chapter 6.

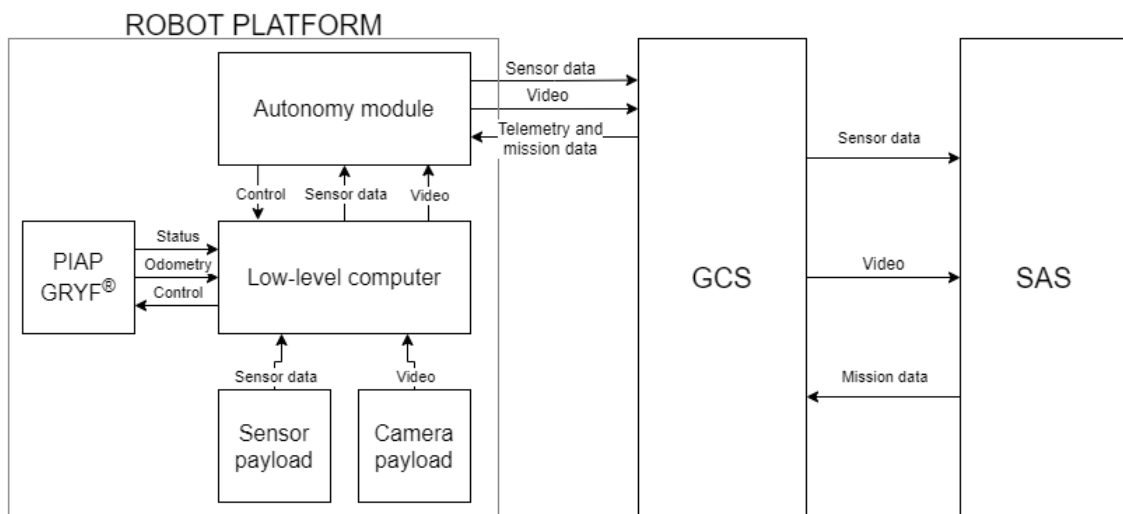


Figure 3 Robot platform general overview

Subchapters in Chapter 4 will focus on description of the robot platform components, as shown in Figure 3, split into hardware and software parts. Unless otherwise specified, all components described in this chapter have been pre-existing before the ASSISTANCE project and have only been adapted for integration with the ASSISTANCE system.

Multiple pre-existing components described in this and the next chapters have been developed in scope of Horizon 2020 CAMELOT project. This includes: autonomy module, autonomy software stack, ROS-JAUS-Bridge.

4.1. Robot hardware overview

4.1.1. PIAP GRYF® platform

The UGV used in the project is based on the PIAP GRYF® mobile robot, produced by Łukasiewicz-PIAP and intended for use in EOD and reconnaissance of terrain and hard to reach places. Detailed specifications of the PIAP GRYF® robot platform have been included in deliverable D4.1.

D4.3 Robots integrated into the system



Figure 4 PIAP GRYF®

Due to its intended use PIAP GRYF® is controlled by the dedicated GCS through the custom wireless link using JAUS standard protocols. JAUS standard defines multiple communication protocols for connecting with remote control systems, JAUS compatibility is described with more details in 5.2. To allow the remote control from other devices, the UGV is equipped with an additional low-level computer, which allows the robot to be controlled through the Ethernet. More details about its functions can be found in chapters 4.2.1 and 4.2.2.

4.1.2. Autonomy module

The autonomy module computing unit is Intel® NUC (Table 1). This mini PC is running Ubuntu 20.04 and its main purpose is to run autonomy software stack. It also handles all communication via JAUS with any JAUS compatible GCS in the same network. Autonomy module is also equipped with Velodyne LIDAR Puck VLP-16, Pozyx UWB radio Creator Tag and GNSS receiver HydraBox with embedded IMU. Those sensors are used for localization and obstacle avoidance.

Parameter type	Value
CPU model	Intel® Core™ i7-7567U
CPU parameters	2 core, 3.50 to 4.00 GHz
GPU	Intel Iris Plus Graphics 650
Memory	16GB SO-DIMM DDR4 2133 MHz
Disk space	512 GB

D4.3 Robots integrated into the system

Table 1 Intel® NUC specifications

The autonomy module captures raw data from low level robot components and inputs them into autonomy stack. This includes both raw data that is used internally as well as data that is forwarded via JAUS to the GCS. Internal data is for example pointclouds from Velodyne, odometry from wheel servos, IMU raw readings, etc. The data is collected via multiple interfaces, most relevant of which is the CAN Bus that provides data such as sensor readings, odometry, manipulator joints position and battery level.

Due to the use of the autonomous navigation, a wireless emergency stop is used in the autonomy module. To connect it to the robot, the base platform was modified to allow connecting the E-Stop module directly to the base motor controllers. This allows to safely stop the robot in case of an emergency, without switching off the power of the computing unit.

4.1.3. Enclosure

Enclosure is a custom designed mechanical component that serves multiple functions, such as holding the computing unit and being a unified physical connector adapter for some payloads.

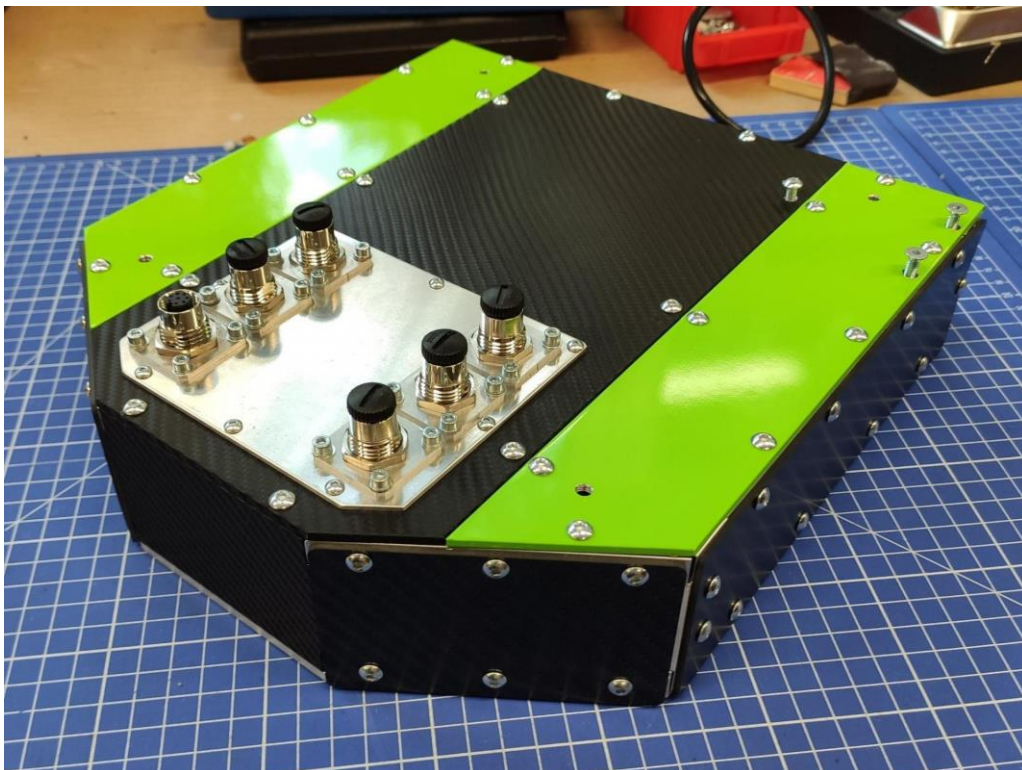


Figure 5 PIAP- GRYF enclosure

The enclosure exposes multiple M12 A-coded 5-pin connectors that are used for connecting all external sensors. These connectors provide access both to the CAN

D4.3 Robots integrated into the system

communication bus to send sensor readings to and the power source for powering sensors.

4.1.4. Communication

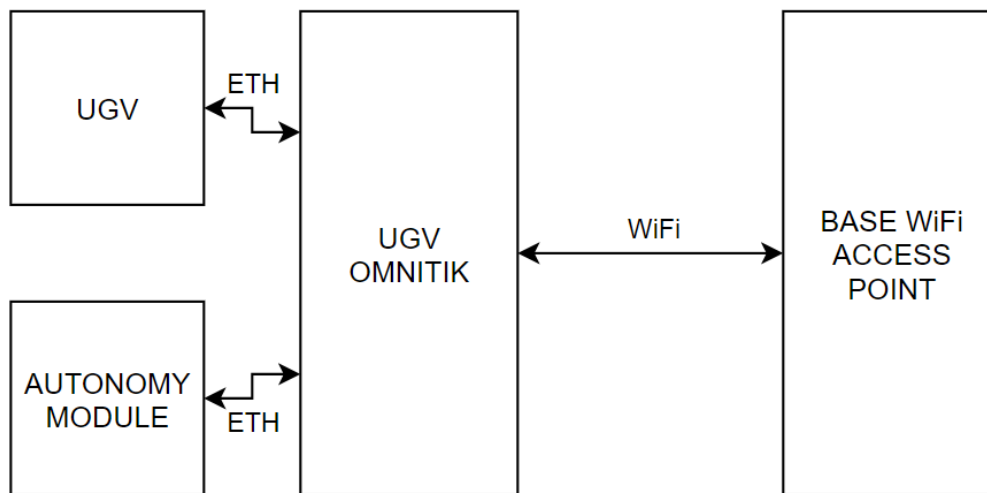


Figure 6 UGV communication architecture

The UGV communicates with the GCS through the WiFi bridge set on two OMNITIK 5 POE AC devices – one mounted on the UGV, connected to the autonomy module and the second one connected to the GCS.

4.2. Robot software overview

4.2.1. Low-level computer

The low-level computer allows the autonomy module to control PIAP GRYF®. To achieve that, two software modules are run on it:

- CAN to Ethernet bidirectional bridge, which allows the autonomy module to control the mobile base, manipulator and sensors connected to the platform CAN bus'
- Video streaming server, which parses the feeds from the cameras connected to the UGV using the additional frame grabber card, encodes it and sends it to the autonomy module.

To increase the security of the solution, the low level computer is connected to the computing unit of the autonomy module via separate subnetwork that is not bridged to any other network. This results in low level computer being accessible only by the autonomy module.

4.2.2. Video server

The video streams from the robot are sent as a unicast stream to the autonomy module. To make them accessible from the SAS, a service has been implemented, which retransmits all of the received video stream packets to the GCS, where they are exposed via RTSP server.

4.2.3. ROS architecture

Robot Operating System (ROS) is an open-source platform for software development and launch of robots. ROS operates on a distributed structure of processes (Nodes) that can be easily grouped into packages and stacks. These packages can be easily shared and distributed. As of today, ROS includes many packages that allow to control and simulate the work of the robot platform. More information describing the ROS specification can be found on the official websites dedicated to this product, i.e. www.ros.org, wiki.ros.org.

In this project, ROS has been used in the autonomy module for communication between the nodes and building custom solutions based on standard elements from the ROS robot navigation package.

The graph shown on

Figure 7 presents the general overview of the main nodes and the communication between them in the autonomy module as well as exemplary ROS-JAUS-Bridge ROS node.

D4.3 Robots integrated into the system

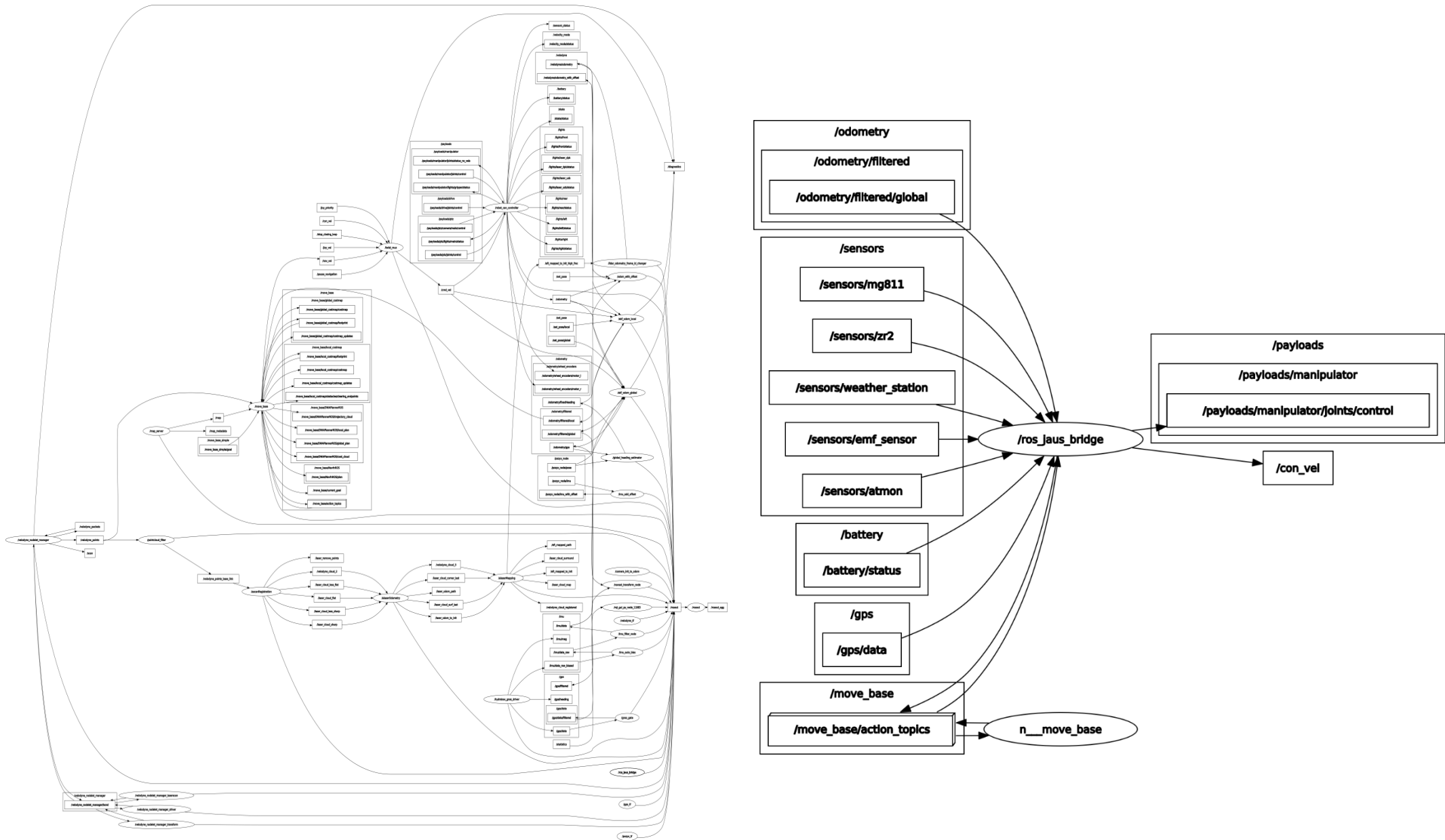


Figure 7 Dependencies between nodes in the autonomy module (left) and exemplary ROS-JAUS-Bridge ROS node (right).

4.2.4. JAUS architecture

Joint Architecture for Unmanned Systems (JAUS) is an international standard of the SAE AS-4 Unmanned Systems Steering Committee that defines communication protocols for unmanned vehicle systems, some of their internal components, and their interaction with operator control stations. More information about the service specifications can be found in the JAUS documentation and websites e.g. sto.nat.int, openjaus.com.

The use of JAUS has many advantages – apart from being a good defined international standard, it opens the possibility of integrating any other UGVs with the JAUS interface implemented (unification of solutions). This approach has allowed to make the ground robot subsystem to be more interoperable as well as platform-independent and required only the development of an additional component that will guarantee the communication compatibility between the SAS interfaces (MQTT, NATS, DDS REST-API) and the UGV interface (JAUS), which is the SAS Adapter, described in more detail in Chapter 5.3.

4.2.5. Project specific optimizations

Payloads configuration modification requires changes to restrictions of joint positions on the manipulator. In ASSISTANCE, due to modular approach and multiple possible payload configurations, instead of multiple static configurations for restricting certain manipulator joints position an additional control interface, based on ROS MoveIt was implemented, which detects and prevents collisions during the path planning stage.

5. GCS

5.1. GCS architecture

5.1.1. General architecture

The robot's Ground Control Station (GCS) is a composition of all hardware and software being the interface between the robot (PIAP-GRYF) and the external system (SAS) and/or the user.

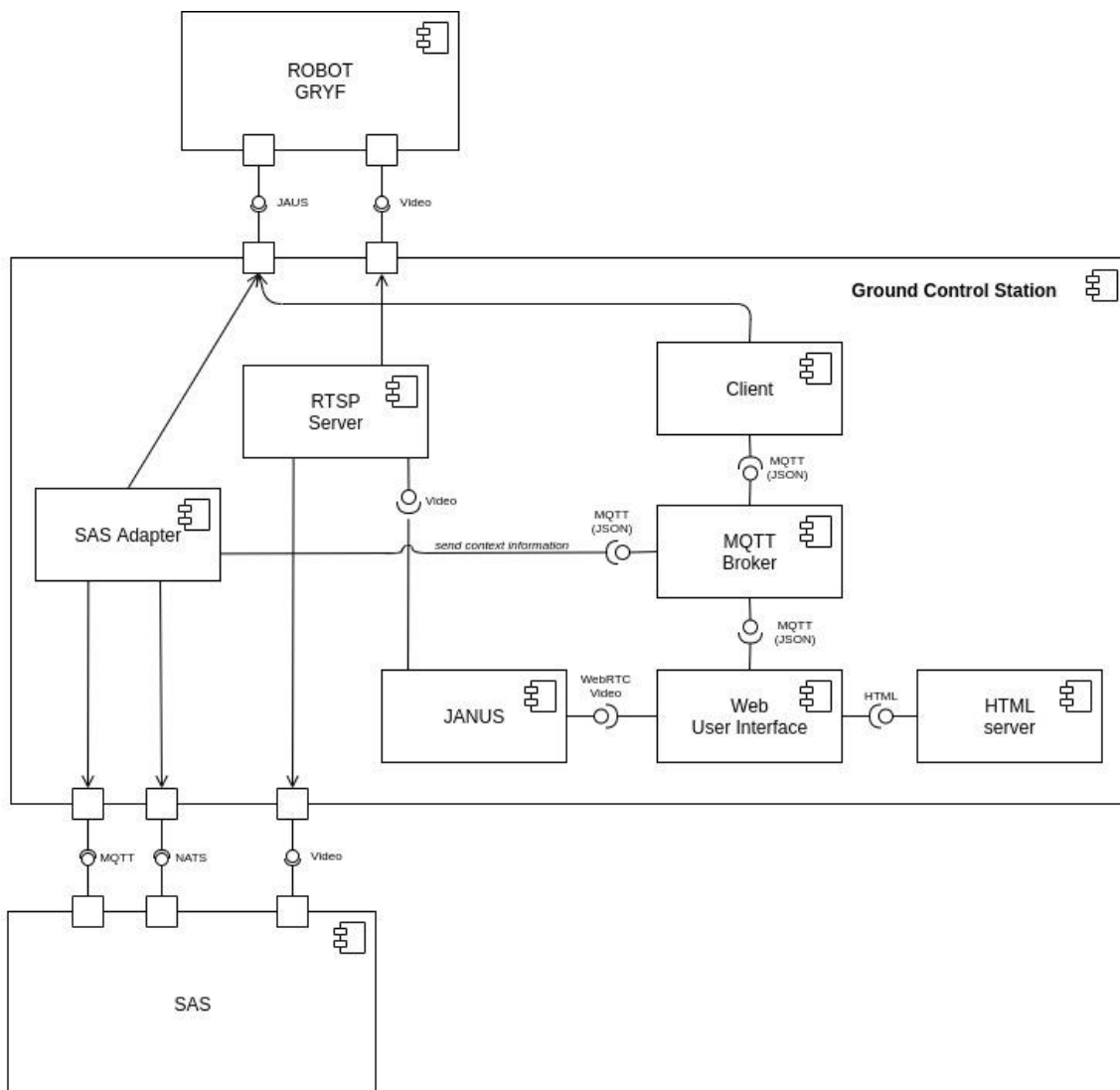


Figure 8 GCS components

The Ground Control Station includes the following components:

- **SAS Adapter** - component working as an adapter ensuring communication with the Sensor Abstraction Service (SAS) layer at more levels. In this module, the received information is processed and converted from/to the JAIUS protocol.

D4.3 Robots integrated into the system

- **Client** - component translating data from the JAUS interface to the MQTT interface operated in the browser. At the same time, the data format is changed from/to JAUS (binary format) - JSON (text format).
- **HTML server** - delivering an application to control the robot. The application was created on the basis of Vue.js framework² made for development of Single-Page Applications. Its library is focused on the view layer only, and is easy to integrate with other libraries or projects.
- **MQTT Broker** – supporting the MQTT protocol. For this purpose, an open source message broker Mosquitto³ has been used. The MQTT protocol provides a lightweight method of carrying out messaging using a publish/subscribe model.
- **RTSP Server** - delivering several media streams simultaneously and independently of each other. This server was created using open source multimedia framework – GStreamer⁴.
- **JANUS** – a general purpose WebRTC server for communication with the browser. Janus consists of the main module (general purpose server) and plugins. The main module provides the mechanisms to configure media communication with the browser. Plugins implement specific functionalities. In the implemented configuration, a plug for video streaming is used.
- **Web User Interface** – application software for robot control.

5.1.2. Web User Interface functional modifications

Modification of the Web User Interface component is related to the idea of the "context". The "context" identifies a specific operation or incident to which all the corresponding objects must be related, such as missions, measurements. This means that no mission can be commissioned and performed if the context is not set. The currently set context is controlled by the SAS Adapter and can be changed after receiving the "context" message. This message can be sent from the SAS or by the operator. In the second case, the operator must be able to read the names of available contexts, select a context from the list and accept it, and/or enter a new context. To enable that remodelling of the Web User Interface was required by adding there a "context manager". The context manager creates a channel between the SAS Adapter and the Web User Interface.

² <https://v3.vuejs.org/guide/introduction.html>

³ <https://mosquitto.org/>

⁴ <https://gstreamer.freedesktop.org/>

This channel is used to send information about the current context and a list of available contexts. Additionally, the Web User Interface has been changed, by adding a widget to it that allows displaying the menu with a list of contexts.

5.2. SAS Adapter

The Sensor Abstraction Service (SAS) is the main element that allows the integration of the ASSISTANCE system elements, including the unmanned vehicles (UAVs or UGVs), with the ASSISTANCE Situational Platform (SAP) and other ASSISTANCE modules. The SAS realizes two-way information exchange – mission related data is sent to the unmanned vehicles (full definition of the mission is included in the next subsection) and the telemetry data, sensor data and images/videos are fed into the ASSISTANCE SAP. In order to integrate the UGV subsystem with the ASSISTANCE system via SAS, which supports different interfaces (i.e. MQTT, NATS, REST-API, DDS) and the format, data structure and logic related to the missions than the UGV (i.e. JAUS) a dedicated software component had to be created that would guarantee the compatibility of these two environments – the SAS Adapter.

The SAS Adapter performs the following tasks:

- *Mission management* – some tasks related to mission are shifted in place and delayed (e.g. measurements at specific points), the SAS Adapter manages this process by running tasks and actions at the appropriate times.
- *Data conversion* between the SAS and the robot interface (JAUS) – the SAS sends and receives data in JSON format, the structure of this data does not translate into one-to-one JAUS format (description of the message structure in JSON format has been defined in the document "SAS database structure and data model" created under WP3 by ETRA).
- *Validation* of the received data – it is checked for the correctness of the structure.
- *Data storage*, some data required by the SAS is not available on the robot or is not transmitted via the JAUS protocol and are stored in the SAS Adapter, as described in detail in the following subchapters.
- *Keeping connection and managing the connection* (ex. request control) between the SAS environment and the robot. If the connection is broken, the SAS Adapter performs the reconnection.

5.2.1. SAS Adapter interfaces

The SAS supports four methods of data exchange: MQTT, NATS, API Rest, DDP. Two of them were selected to be supported by the SAS Adapter, i.e. MQTT and NATS.

5.2.2. MQTT interface

This messaging system is based on a method of publication and subscription of topic (similar to ROS topic). Messages are queued and sent asynchronously and the publisher and subscriber do not have any information about each other, except for information about the topic.

5.2.2.1. Resource

The message “Resource” is one of the two primary messages sent from the SAS Adapter to the SAS. This message is periodically published from the moment when SAS Adapter receives information about the actual context.

Important.

Receiving the context is the condition to start sending the data (information about resources, information about the state of sensors). The context can be entered by the GCS operator or arrive from the SAS.

As many of the variables in this message have constant values, most of the information sent in this message is taken from the configuration file when the SAS Adapter is launched.

The message contains the following fields:

Field	Description
_id	A resource unique identifier - in our case "gryf" string.
type	A type of resource identifier - in our case "unmanned".
subtype	A subtype of resource - in our case "ground".
hierarchy	A position of the unit in its organization - in our case this value is empty because only one robot is used and this generates a clear structure.
plate	Vehicle plate - in our case "L-PIAP_GRYF".
workingAltitudeMinimum	Minimum altitude in meters at which the unmanned vehicle can work – in our case it is value <0, 9999>.
workingAltitudeMaximum	Maximum altitude in meters at which the unmanned vehicle can work – in our case it is value <0, 9999>.
operationSpeed	Speed in m/s at which the unmanned vehicle moves in normal – in our case 1 m/s.

D4.3 Robots integrated into the system

climbSpeed	Speed in m/s at which the unmanned vehicle lifts in, in normal – in our case it's 0 m/s because the robot only moves in the horizontal surface.
context	Current context identifier linked to the robot.
mission	Identifier of the mission (if any) to which the last status is related – in our case at the beginning it's an empty string, the value changes when a mission is received.
remainingAutonomy	Approximate minutes of battery life remaining – in our case it's 120 minutes.
geometry	GeoJSON geometry object to represent the last known location of the resource – in our case the robot position (the position value is periodically received from the robot in the JAUS "ReportGlobalPose" message).
height	Height above ground at last status – in our case it's 0 m all the time.
speed	Vehicle speed at last status – in our case it's the current speed of the robot (the speed value is periodically received from the robot in the JAUS "ReportTravelSpeed" message).
attitude	Attitude (pitch, roll, yaw) at last status – in our case it's [0,0, current yaw of robot] (the source of the value is the same JAUS message as for the geometry value).
cameras	Properties and last status of the resource cameras– in our case these values are loaded from the configuration file and describe the video stream parameters, camera position, camera offset from the resource's GPS.
timestamp	Date in ISO 8601 format on which the resource is inserted/updated e.g. 2020-01-01T10:00:00.000Z.

Table 2 Resource message

5.2.2.2. Sensor

Similarly to the "Resource" message, a lot of information about the sensors list and sensors properties is stored in the SAS Adapter configuration file. The reason is the same as with "Resource" messages – these variables have constant values or JAUS does not support their transfer.

The message contains the following fields:

Field	Description
_id	sensor unique identifier (ex. AirPressure, NeutronsCount)
type	type of sensor (ex. pressure, gray_per_h)
context	identifier of the context to which the last measurement is related
mission	identifier of the mission (if any) to which the last measurement is related
geometry	location where the last measurement was taken (it is GeoJSON object) – in our case it's the robot position (the position value is periodically received in the JAUS "ReportGlobalPose" message from the robot)
measurement	last measurement data, this object contains the following fields: <ul style="list-style-type: none"> • <i>metricFeature</i> - identifier (id) of the metric feature (ex. AirPressure, NeutronsCount) • <i>unit</i> - measurement units (ex. m/s, keV), this value is periodically received from the robot in the JAUS "ReportSensorData" message) • <i>value</i> - measurement value (a number), this value is periodically received from the robot in the JAUS "ReportSensorData" message)
timestamp	date in ISO 8601 format on which the sensor is inserted/updated

Table 3 Sensor message

5.2.3. NATS adapter

This technology is also based on a subscription/publication model, such as MQTT, but it offers more options for controlling the flow of messages. For example, NATS offers the following properties:

- NATS Streaming (includes: enhanced message protocol, message/event persistence, at-least-once-delivery, publisher rate limiting, rate matching/limiting per subscriber, historical message replay by subject, durable subscriptions),
- ACKs (published messages require acknowledgment),
- sequence numbers (a sequence number is added to the message),

D4.3 Robots integrated into the system

- queue groups (it is used for balanced message delivery in a group of subscribers).

In addition, NATS offers another important message distribution method – a request-reply mechanism, which is used by the SAS Adapter when it starts. During program initialization, application sends a request about a current list of contexts and missions, after which during standard operations the subscription and publication mechanism is used.

5.2.3.1. Context

Context identifies a specific operation or incident to which all the corresponding objects must be related, such as missions, measurements, evacuation routes, etc.

The message contains the following fields:

Field	Description
_id	context unique identifier
description	short description of the mission
context	brief description of the incident or operation
geometry	geographic data as a GeoJSON geometry object that delimits the area of action of the context
start	date in ISO 8601 format on which the context begins
timestamp	date in ISO 8601 format on which the context is inserted/updated

Table 4 Context message

5.2.3.2. Mission

A mission is a set of actions undertaken by an indicated object (unmanned vehicle, unmanned plane) in specified places and at specific times, which results in a set of measurement data (sensor data, camera image). These assumptions translate into the structure of the message.

This message contains the following fields:

Field	Description
_expired	"bool" tag informing if the mission is still valid
_id	it is mission unique identifier (ex. gryf)

D4.3 Robots integrated into the system

description	short description of the mission
context	identifier of the context to which the mission is related
resource	identifier of the resource to which the mission applies
geometry	GeoJSON geometry object to represent the path that the resource must follow, in the SAS Adapter this field is transformed into a "GlobalWaypointList" message.
actions	actions to be taken by the resource, it is an array with sensor actions items such as: <ul style="list-style-type: none">• identifier of the sensor that the action involves,• index of the point which the action will take place,• identifier of the action to be taken (ex. COmeasurement),• optional action parameters.
start	date in ISO 8601 format on which the mission begins
end	date in ISO 8601 format estimating the end of the mission
timestamp	date in ISO 8601 format on which the mission is inserted/updated

Table 5 Mission message

When the SAS Adapter connects to the SAS, it gets the current mission list. If this list includes a mission which resource field contains the "id" of the supported robot, the mission is sent to be executed. The same procedure is performed when the mission list is updated while the program is operating.

5.2.4. Sensor action

"Sensor actions" is a specific modification of the UGV and GCS software as part of the robot adaptation to the ASSISTANCE environment. The main goal of this modification was to reduce the robot system power consumption. Originally, all installed sensors were turned on along with the robot base. Currently, this process is controlled and each sensor can be turned on and off separately.

On a GCS level, an additional service was included to the JAUS protocol that allows the control of sensors. This service has been implemented in the ROS-JAUS-Bridge, the user interface and the SAS Adapter.

Implementation of this functionality in the SAS Adapter allows the execution of "actions" (an element of the structure of the received mission) regarding the sensors, i.e. taking measurements only in specific places on the mission route.

5.3. JAUS compatibility

5.3.1. Implemented services

The implemented services are presented below, with an overview of their functionality.

Core group

Implementation of services from this group enabled functionalities concerning prioritization of access to the control of the robot components.

- *Management* – provides a state machine for component life-cycle management.
- *Discovery* – supports the discovery of both legacy components and new components. The discovery process is performed at two levels at the node and subsystem level.
- *Discovery Client* - service which only implements the "client" side protocol of the Discovery process.
- *AccessControl* – offers a basic interface for acquiring preemptable exclusive control to related services that utilize this function. Once the exclusive control is established, the related services shall only execute commands originating from the controlling component.

Mobility group

Implementation of services from this group made enabled functionalities related to the realisation of the mission.

- *PrimitiveDriver* – provides basic platform mobility control without implying any particular platform type.
- *GlobalPoseSensor* – provides the global position and orientation of the platform.
- *GlobalWaypointDriver* – allows setting a single target waypoint and implementing platform movements to it.
- *GlobalWaypointListDriver* - realizes the movement of the platform along a given series of target waypoints.

Environment group

Implementation of services from this group enabled functionalities concerning the video signal transmission.

D4.3 Robots integrated into the system

- *DigitalVideo* – provides access to the capabilities and configuration of the digital visual sensor, allowing the controlling component to set the visual sensor to a particular operational profile.
- *VisualSensor* – provides access to the basic capabilities and configuration of a visual sensor, allowing the controlling component to set the visual sensor to a particular operational profile.

Interoperability Profile (IOP) group

Implementation of services from this group enabled functionalities related to video signal transmission (the same as in the case of implementation "Environment group").

- *DigitalResourceDiscovery* provides mechanisms to find units that send multimedia data (video streams, picture files).
- *PlatformState* provides mechanisms to find units that send multimedia data (video streams, picture files). It allows monitor and control the status of all components on the platform.

Manipulator group

Implementation of services from this group enabled functionalities concerning the robot arm control.

- *PrimitiveManipulator* – the low level interface to a manipulator arm. This service, which allows each joint of the robot arm to be independently controlled.
- *ManipulatorSpecification* – used to describe a manipulator arm. When queried, the service will reply with a description of the manipulator's specification parameters, axes range of motion, and axes velocity limits.

Unmanned Ground Vehicle (UGV) group

Implementation of services from this enabled functionalities related to energy level management

- *PowerPlantManager* – provides the means to control power source (battery) of the vehicle.

Extra group

Implementation of services from this group enabled functionalities related to sending information from the robot sensors to the SAS.

- *SensorData* – this service is our own extension to the JAUS standard and allows transferring data from the robot's sensors.

5.4. Data streams

5.4.1. JANUS gateway

JANUS gateway (Figure 9) is used to receive the video stream from an RTSP server and forward it to a web browser using WebRTC (Web Real-Time Communication), an HTML5 standard developed by the World Wide Web Consortium (W3C) for real-time P2P communication.

While the plugins provide specific functionality, the JANUS core is responsible for everything that is related to the WebRTC itself. Specifically, the core ensures that a WebRTC PeerConnection can be correctly negotiated and established with interested users, and that this PeerConnection can be used to send and receive media in a way that the WebRTC users can understand.

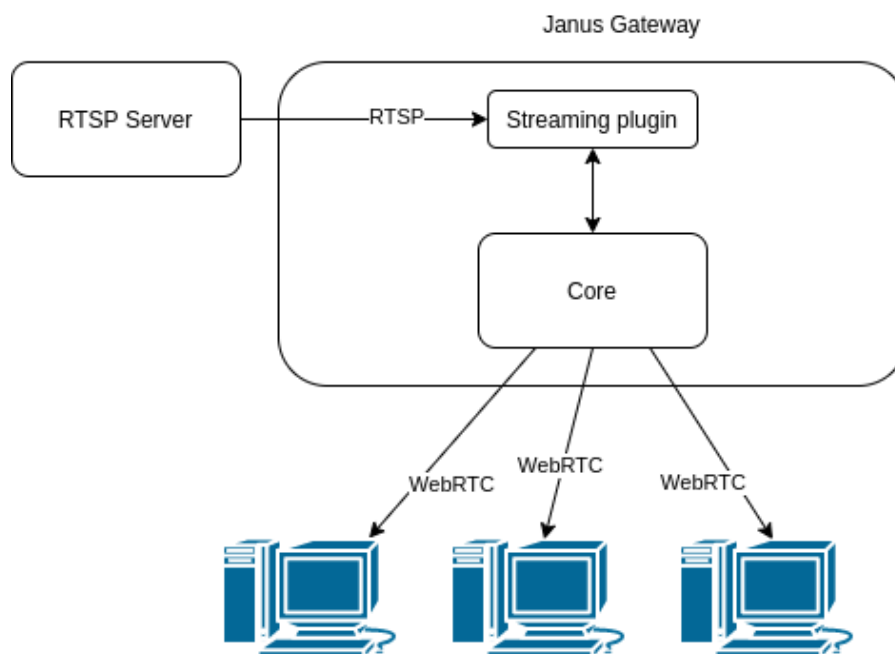


Figure 9 Janus Gateway

5.4.2. Sensor data

The structure of the data flow from sensors is presented in Figure 10.

D4.3 Robots integrated into the system

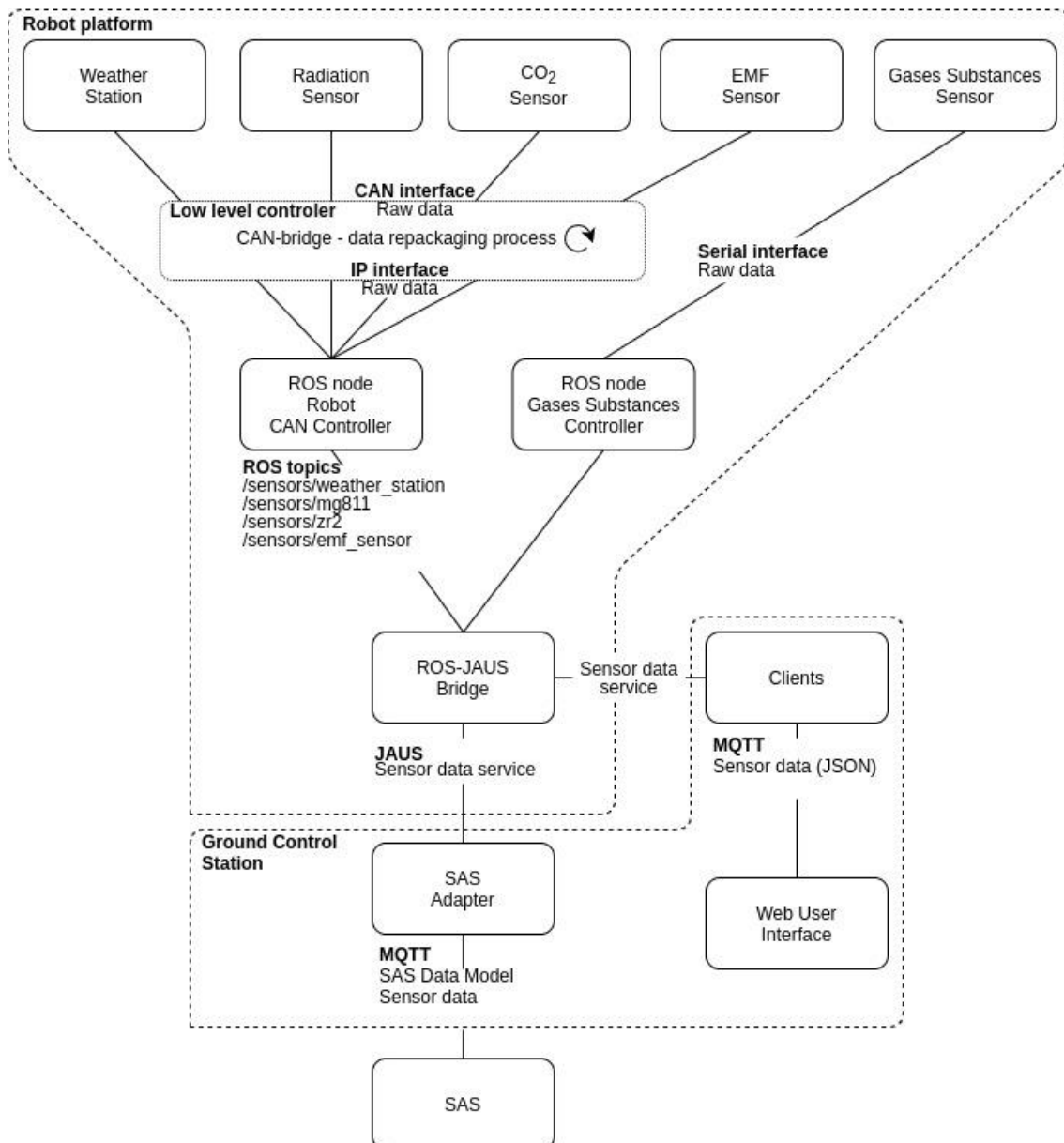


Figure 10 Sensor data exchange information

The physical medium on which the measurement data from the sensors are sent to the low level computer is the CAN interface. In the low level computer, the data is only repackaged into UDP frames and sent to the autonomy module, where it received by the ROS node - "Robot CAN Controller". This program validates the data and sends it to the appropriate ROS topic. The exception is the Gas Substances Sensor, data from this sensor are validated and sent on ROS topic by another program. This is due to the fact that communication with the gas sensor is based on a serial interface.

In the next step, the measurement values are received by the ROS-JAUS-Bridge where those measurements are paired with additional required metadata, such as:

- extended name of the sensor (e.g. "Gamma Detector"),
- name of data (e.g. "Neutrons count"),

D4.3 Robots integrated into the system

- metric (e.g. "keV"),
- the range of data (e.g. value from "0" to "6500"),
- value type (e.g. "uint16").

Sensor data prepared by the ROS-JAUS-Bridge are used by the SAS Adapter, where they are converted to JSON format and formatted according to the specification included in the data model. In this form, the data is sent to the SAS.

The second receiver of data from the ROS-JAUS-Bridge is the Web User Interface. To receive the data, the browser uses the MQTT protocol and the data exchange format – JSON. The conversion of protocols and data formats is done in the module "Clients".

In the case of the SAS and Web User Interface, the data is in JSON format, but the data structure is different. This is the result of different requirements and the fact that the interface between the browser and the "Clients" program was created earlier.

5.5. User interface

The GCS provides a web user interface for operation of one or more robots. This user interface has been developed prior to the ASSISTANCE project. However, due to project requirement ROB_017 “Control system should be user-friendly” being rated as one of most important requirements for the robot, some additional effort has been taken to enhance user experience in this web user interface. Enhancements include:

- Adding serializable/loadable/editable layout layers
- Support for additional themes (ex. Dark theme)
- Multiple sorting/filtering capabilities
- Friendly names for JAUS addressing
- Engine update to Vue.js 3 for increased performance
- General user experience enhancements

For map visualization the GCS uses map data provided by OpenStreetMap® which is licensed under Open Data Commons Open Database License available at <https://www.openstreetmap.org/copyright>. Copyright © OpenStreetMap contributors.

5.5.1. Requirements

The system requirements for this application were:

- Operating system: Linux Ubuntu 20.04.
- Internet browser: Chrome, Firefox.
- RAM: 4GB or more.
- Display: Minimum Resolution: 1024 x 768 in 16-bit color,
- Recommended Resolution: 1920 x 1080 in 32-bit color.

5.5.2. General view

The most important functions that can be performed via the user interface are: robot autonomous driving via waypoints, manual driving in teleoperation mode, displaying data from sensors, displaying video streams from registered cameras and showing position of all connected robots.

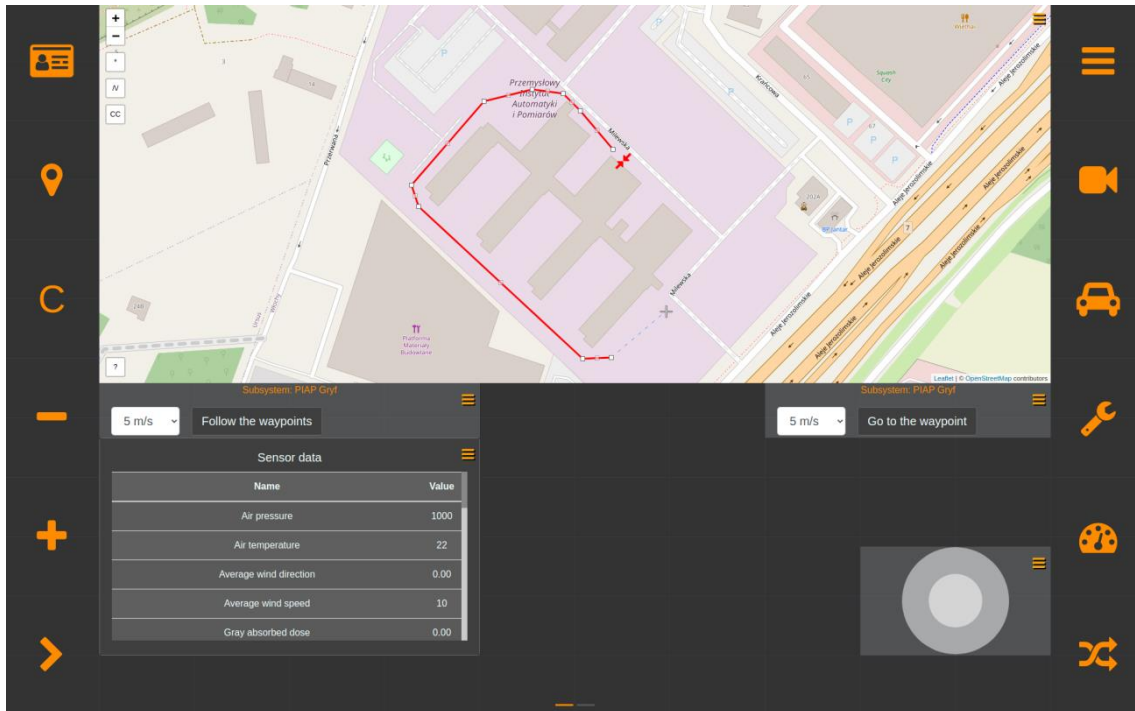


Figure 11 GCS main view

To control the robot in teleoperation mode, the widget of the service responsible for it needs to be loaded (Primitive Driver) that enables to take control over this service (Figure 12).

The robot can perform a user-defined or external mission (autonomy mode), received by the SAS Adapter.

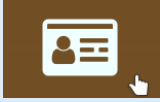

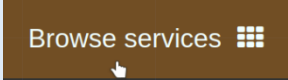

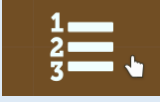



If the robot performs an external mission received by the SAS Adapter, there is an option of interrupting it by taking control of the robot (Figure 12).

D4.3 Robots integrated into the system

Address	Controller	Services	Action
3.1.1	Not controlled	PlatformState	Request control
3.1.2	4.1.1	PrimitiveDriver	Release control
3.1.3	Not controlled	PrimitiveManipulator, ManipulatorSpecification	Request control
3.1.4	Not controlled	GlobalPoseSensor	Request control
3.1.5	Not controlled		Request control
3.1.6	Not controlled	GlobalWaypointDriver, GlobalWaypointListDriver	Request control
3.1.7	Not controlled		Request control
3.1.8	Not controlled	VisualSensor, DigitalResourceDiscovery, DigitalVideo	Request control

Figure 12 Access Control

Main view icons:

Icon	Description
	Open Access Control window.
	Add new view
	Browse all services
	Choose the second menu
	Open Context management window
	Open the layer editor window
	Select the next view
	Choose a specific view

D4.3 Robots integrated into the system

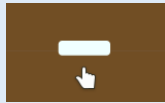



	Delete current view
	Sort services by camera type
	Sort services by manipulator type
	Sort services by driver type

Table 6 GCS main view icons descriptions

5.5.3. Waypoint editor

For the autonomous driving mode, an option of defining a list of waypoints on the map and their subsequent editing, i.e. removing and adding intermediate waypoints, is available.

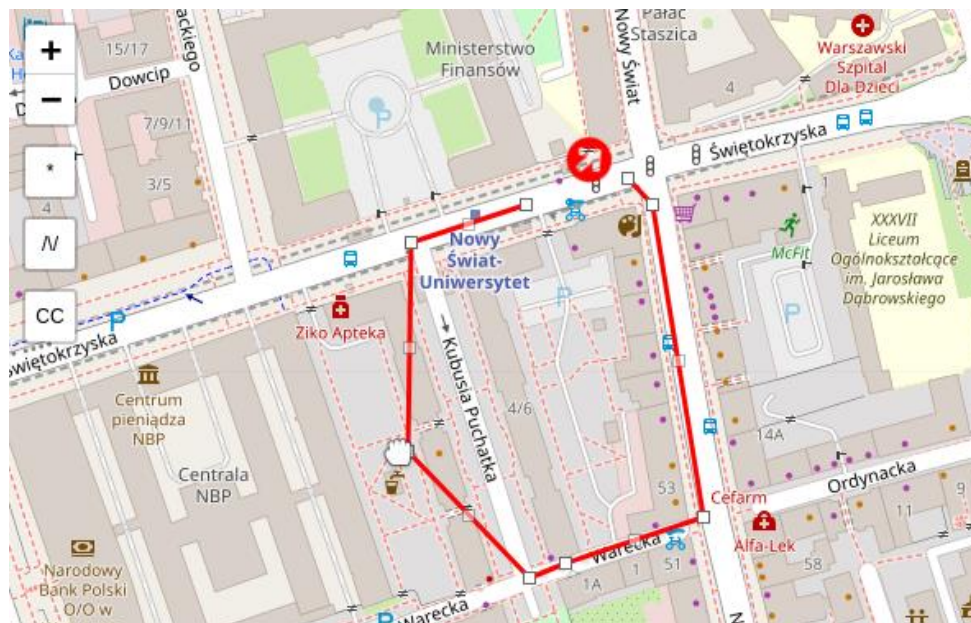
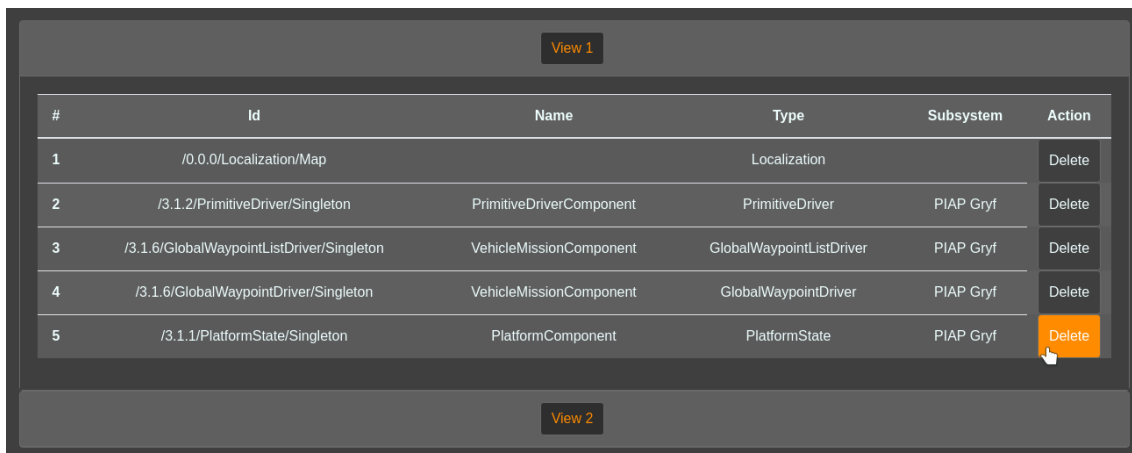


Figure 13 Editing waypoints on map

5.5.4. Layer editor

The view can be adjusted to the user needs by adding or removing widgets to it. Many views are available with an option to switch between them.

D4.3 Robots integrated into the system



The screenshot shows a 'Layer editor' window with a table of components. The table has columns for '#', 'Id', 'Name', 'Type', 'Subsystem', and 'Action'. There are five rows of data. The 'Delete' button in the last row is highlighted with a mouse cursor.

#	Id	Name	Type	Subsystem	Action
1	/0.0.0/Localization/Map		Localization		Delete
2	/3.1.2/PrimitiveDriver/Singleton	PrimitiveDriverComponent	PrimitiveDriver	PIAP Gryf	Delete
3	/3.1.6/GlobalWaypointListDriver/Singleton	VehicleMissionComponent	GlobalWaypointListDriver	PIAP Gryf	Delete
4	/3.1.6/GlobalWaypointDriver/Singleton	VehicleMissionComponent	GlobalWaypointDriver	PIAP Gryf	Delete
5	/3.1.1/PlatformState/Singleton	PlatformComponent	PlatformState	PIAP Gryf	Delete

Figure 14 Layer editor

5.5.5. Services

Services are JAUS standard encapsulations of specific functionalities of the robot system. Each implemented service (see chapter 5.3.1 for the complete list), that is running is shown in the UI. Each of these services has a service-type specific widget to help the operator to control the robot.

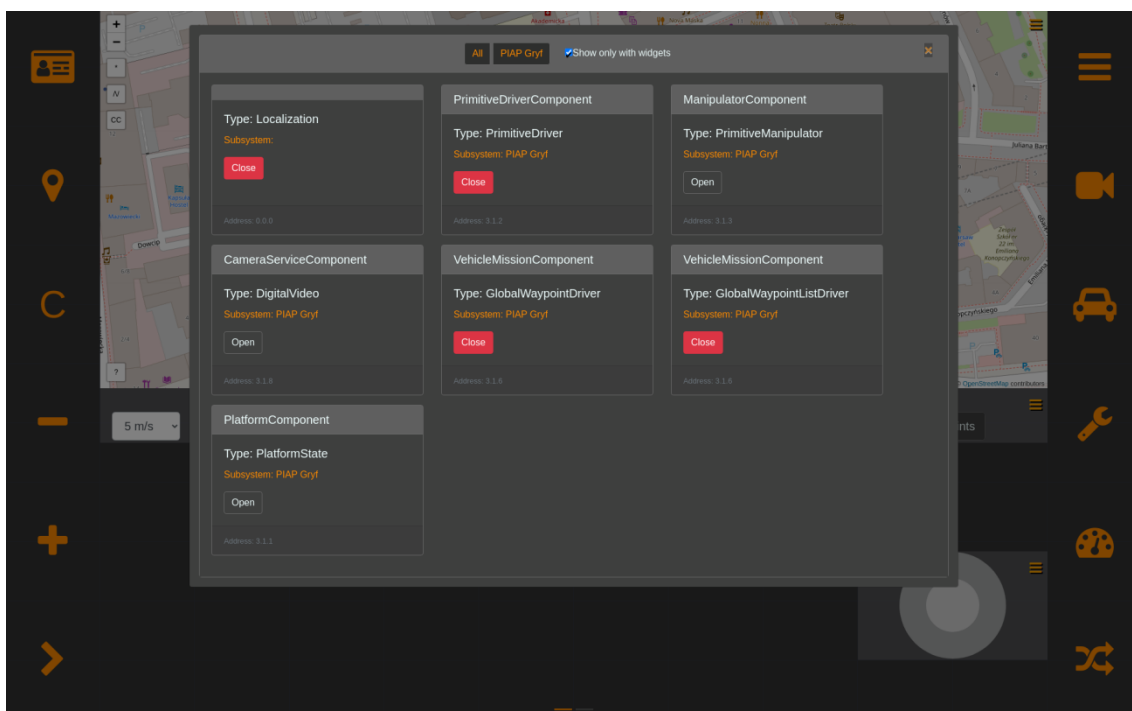


Figure 15 All services view with default settings

Each service is represented by the option window.

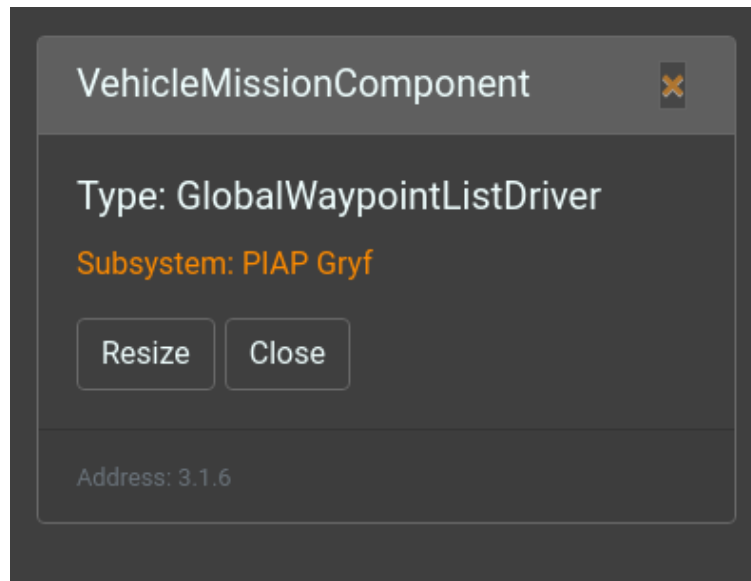


Figure 16 Example service card

The card contains basic information about the service, which are:

- VehicleMissionComponent: name of the service.
- Type: type of the service.
- Subsystem: name of the vehicle or the GCS.
- Resize button: is present when the service has the widget and is used to move or change size.
- Open/Close button: is present when the service has a widget, it's used to add/remove from the current view.
- Address: JAUS address containing subsystem id, node id and component id.

The services can be sorted by:

- Widget: services only with widgets
- Type: service type
- Subsystem: name of the vehicle

5.5.6. Context management

As mentioned in 5.1.2 the contexts can be managed by selecting one from the list (Figure 17).

D4.3 Robots integrated into the system

Resource

Name: gryf

Context: CTXT20200101100000

#	Id	Description
1	CTXT20200101100000	Terrorist attack in The Sydney Opera House
2	ETRA_TEST_CONTEXT	Context created for testing
3	CTXT20210304_CATEC_TEST	CATEC first test sendind telemetry and streaming video during a real flight
4	TestIntegrationPIAP_1	TestIntegrationPIAP_1
5	TestIntegrationPIAP_2	TestIntegrationPIAP_2
6	TestIntegrationCATEC_1	TestIntegrationCATEC_1
7	TestIntegrationCATEC_2	TestIntegrationCATEC_2
8	TestIntegrationCATEC_3	TestIntegrationCATEC_3
9	TestIntegrationCATEC_4	TestIntegrationCATEC_4
10	CHEM001	
11	TestIntegratonPIAP5	TestIntegratonPIAP5
12	CHEM0099	TEST_PLUMES_UPV
13	TEST_UPV	TEST_PLUMES_UPV

List of available contexts

Figure 17 Context assignment component

6. Payloads

Different emergency response scenarios impose different requirements for the UGV. To adapt it to the needs of each mission, it can be equipped with various payloads – devices providing additional data or functions. Analysis of the ASSISTANCE use-case requirements and selection of specific devices as payloads for the UGV was described in detail in D4.1.

All of the payloads are designed in a modular way – most of them share the same electrical, communication and mechanical interface (described in 6.1.1, 6.1.2 and 6.1.3), which allows an easy reconfiguration of the UGV.

Available payloads for the UGV are:

- Sensors:
 - Vaisala WXT-520 weather station
 - ZR-2 radiometer,
 - ATMON FL gas analyzer,
 - MG-811 CO2 concentration sensor,
 - Electromagnetic field intensity sensor,
- Cameras:
 - Four RGB cameras
 - Thermal imaging camera
- Tools and other:
 - CUTLANCA cutting extinguisher

The following subchapters present the most crucial developments and adaptations with regards to the payloads layer of the UGV.

6.1. Common elements

To improve the reconfigurability of the UGV, most of the platform payloads share the same electrical, physical and communication interface, called a common sensor interface. Exceptions are:

- PIAP GRYF® cameras – these are embedded in the base platform and use their own custom connectors and interfaces.

D4.3 Robots integrated into the system

- Thermal imaging camera – as it couldn't use the same communication interface due to the much higher bandwidth in comparison to other payloads, it uses the Ethernet instead.

6.1.1. Common electronic interface

Most of the sensors are plugged in to the platform using the M12 A-coded 5-pin connectors. All of these devices are powered through it from the UGVs battery (22-32V) and communicate via the CAN interface.

6.1.2. Common communication interface

All of the payloads using CAN interface are communicating with the UGV using the CAN open-based protocols. Each of the sensors has its unique ID, all of them can be controlled using the Network Management Protocol, powered on and off. This allows a more efficient management of the power use as all of the sensors can be turned on only when required.

6.1.3. Common physical interface

Most of the payloads share the same mechanical interface – they are mounted on the UGV by the quick release NATO rail clamp. The EMF sensor, due to its measurement characteristics can also be mounted in the gripper using the integrated gripper recess. Both of the mounting options are shown on Figure 18.

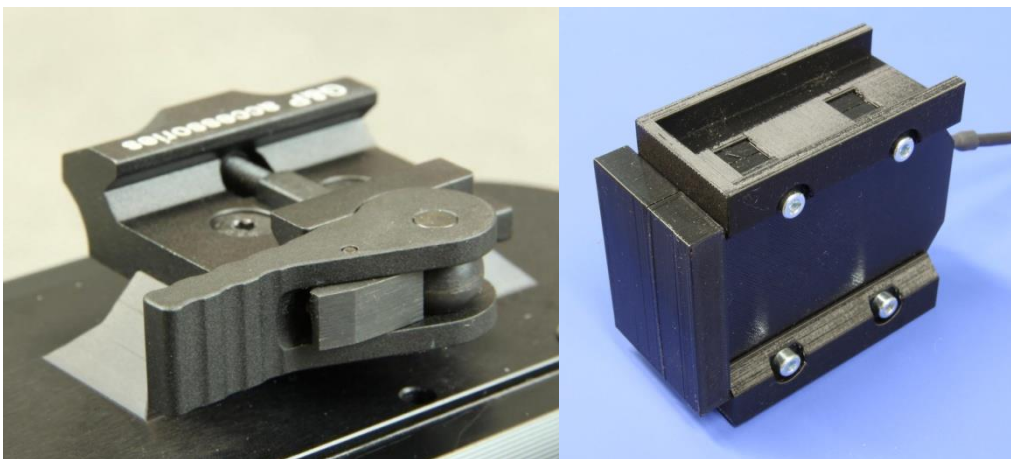


Figure 18 Quick release NATO rail clamps and recess for gripper

6.2. RGB Cameras

PIAP GRYF®, on which the UGV is based, is by default equipped with four RGB cameras:

- Front camera, an inseparable part of the base, facing directly to the front of the robot. It is paired in additional IR lights, which can be turned on to improve the visibility in poor lighting conditions,

D4.3 Robots integrated into the system

- Rear camera, an inseparable part of the base, facing directly to the back of the robot,
- Gripper camera: detachable camera, by default mounted on the small NATO rail over or under the gripper,
- PTZ camera: detachable camera, mounted on the manipulator with an additional link that can be set in desired position. Equipped with lens with a zoom up to 30x and additional lights that can be turned on to improve the visibility in poor lighting conditions. Fitted on a motorized base enabling remote control of the pan and tilt.

All of them output PAL video streams, which further processing is described in chapters 4.2.1 and 4.2.2.

6.3. Thermal camera

The thermal camera available on the UGV is presented on Figure 19.



Figure 19 Thermal imaging camera, mounted on the UGV

6.3.1. Initial research

In order to provide the most adequate and efficient device for thermal imaging with the use of robot, some initial review of the thermal imaging cameras has been performed, which resulted in a list of requirements that was used as a basis for the market research:

- resolution: from 320x240 to 640x480; lower resolution would limit the camera usability, higher resolution would require use of the camera with the cooled sensor, which would drastically increase the cost of it,

D4.3 Robots integrated into the system

- embedded mechanical shutter: it must be used in thermal cameras with uncooled sensors, adding it externally unnecessarily complicates the integration,
- lens: 35-85 FOV, wider angle lens would add a significant distortion, camera with a tighter lens would be less practical on the UGV, especially when operated in buildings,
- protection level fit for the use in an emergency situation,
- compliance with power and communication interfaces available on the UGV,
- manufacturers from the European Union were preferred: ordering high resolution thermal cameras from the USA is subjected to regulations that either limit the maximum framerate or require fulfilling additional formalities,

Two different options were considered:

- using an off-the-shelf thermal camera compliant with all the requirements,
- using a thermal camera module and designing the electronics and enclosure to fit the requirements.

None of the available off-the-shelf thermal cameras were fully compliant with the requirements and integration of any of them would require significant amount of engineering work, usually a design of additional electronics and enclosure for them. They are also more expensive than thermal camera modules with comparable image parameters.

Due to that, a decision has been made to buy a thermal imaging camera module and to integrate it. After the market search, Etronika KTL module was selected. It met all of the requirements and offered the best image quality of a few tested cameras from the similar price range.

6.3.2. Hardware

The thermal module outputs an uncompressed video, which has to be encoded before sending it further. It is a very hardware-intensive task, which usually is done with the help of dedicated encoders. These are usually found in SoCs with relatively powerful CPUs or implemented in the FPGAs. Using either of these on a custom module requires a significant time spent on the design, due to the strict constraints related to the high-speed bus routing, power supply requirement and thermal management.

Due to the relatively small resolution (in comparison with standard RGB cameras) of the processed video stream, alternative solution, which would limit the hardware development time, was considered – use of the high-end microcontroller with embedded MJPEG encoder.

D4.3 Robots integrated into the system

This idea was tested on the NUCLEO-H755ZI-Q development board with the STM32H755ZI microcontroller. Video feed was received from the camera in the BT.656 format, encoded using the embedded MJPEG encoder and then sent through the Ethernet interface as a multicast transmission. The results were satisfying – video was streamed in full resolution and frame rate, with no significant drop in image quality (no perceivable difference in normal use) and with a bitrate fit for the use through the wireless network.

After the test, the design of the custom electronics for the thermal camera had started, meeting these requirements:

- Use of the STM32H755ZI or other comparable MCU: hardware MJPEG encoder, DCMI and Ethernet interfaces and at least 1Mb of SRAM are required.
- Powered by the Power over Ethernet – this improves the ease of use, as only one cable would need to be connected to the camera to use it.
- Size of the board comparable to the dimensions of the thermal camera module – it would make the mechanical integration easier.
- Design should be reconfigurable, to make to possible future changes easy.

Final design was split into two boards. This made the electronics smaller than the outline of the thermal camera and modular – if the camera would have to be powered in a different way or transmit the video through the USB, only the first PCB would have to be changed. The architecture of the electronics stack is shown on the Figure 20 and finished PCBs are shown on the Figure 21.

To provide an appropriate protection for use in emergency situations for both camera module and electronics, a custom made enclosure with the IP65 ingress protection rating and additional germanium glass to protect the lens (standard glass cannot be used, as it doesn't transmit infrared light) was ordered from the manufacturer of the thermal camera module.

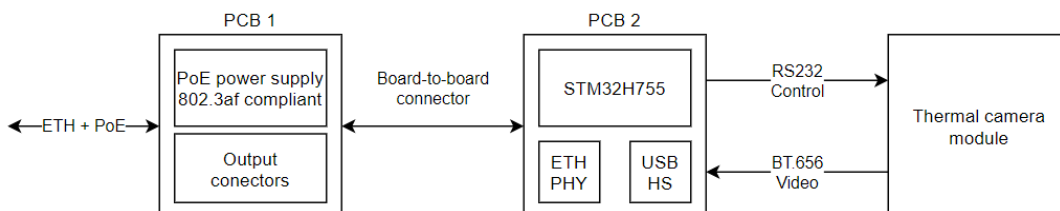


Figure 20 Thermal camera electronics architecture

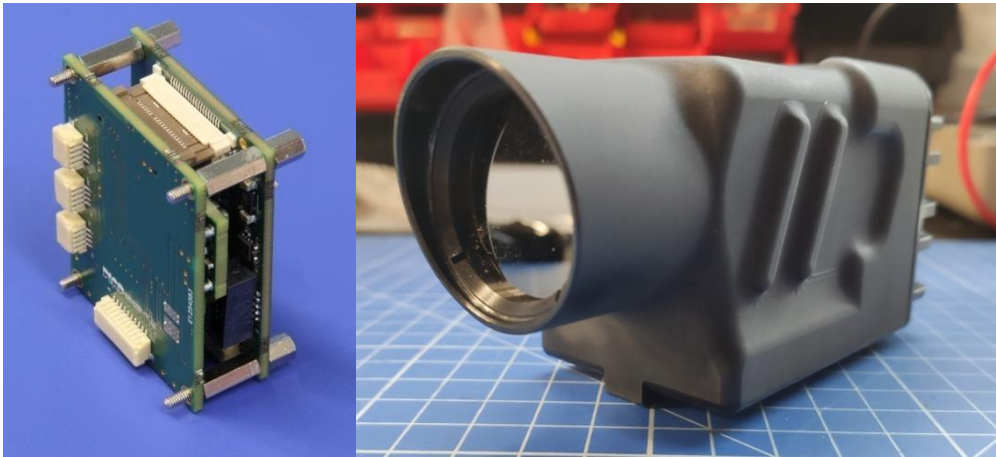


Figure 21 Finished thermal camera electronics and custom made enclosure

6.3.3. Video transcoding

To additionally reduce the wireless bandwidth used by the thermal camera, its video stream can be transcoded from MJPEG to H.264 on the computer embedded in the autonomy module. Initial tests were done using the CPU-based encoding, which caused a significant decrease in the bitrate of the video stream: from 4 Mbps to around 1.5 Mbps, without any perceivable difference in quality.

The CPU used in the autonomy module computer has an integrated GPU, with an embedded low power H.264 hardware encoder. To reduce the strain on the CPU and improve the power efficiency, a script used during the initial tests was modified. As a result, the additional usage of the CPU by the transcoding was reduced (from around 50% to 3% of one core), while keeping previously mentioned decrease in the bitrate.

6.3.4. Testing

Thermal camera was used during multiple indoor and outdoor test sessions, in both cases it provided video of good quality, allowing the easy detection and recognition of hot spots, people and other objects. Figure 22 shows two example photos taken with the camera.



Figure 22 Example photos taken with thermal camera

6.4. ATMON FL gas analyser

ATMON FL is a mobile measurement system for gaseous and powdery air pollutants. It has been originally designed for UAVs, however, it also offers modes for stationary and ground operations. This sensor is commercially available on the market and for project demonstrators it will be supplied by CNBOP partner.

This sensor has its own internal battery and provides data via radio. Radio is supplied together with main unit, and connects using USB. For this reason the robot chassis design had to include the additional USB ports as well as provide enough surface space to allow the main unit to be mounted.

ATMON FL comes with the Windows GUI application for data visualization. Due to the platform limitation – both robot platform and GCS are running on Linux – dedicated Linux driver has been developed. Protocol used for providing data had to be documented using reverse engineering technique. The reverse engineering of protocol was possible only due to no encryption being used when reporting sensor readings.



Figure 23 ATMON FL main unit

6.4.1. Measured substances

ATMON FL provides modular substance detection and measurement sensors. Two sensors are fixed: PM2.5 (particulate matter with a diameter of 2.5 μm or less) and PM10 (particulate matter with a diameter of 10 μm or less). Other than those two, up to 4 sensors can be attached to ATMON FL. List of possible measured substances with units ranges of measurement and accuracy are shown in Table 7.

Module	Substance	Units	Range	Step
PM	PM2.5	$\mu\text{g}/\text{m}^3$	0-999	15
PM	PM10	$\mu\text{g}/\text{m}^3$	0-999	15

D4.3 Robots integrated into the system

SO2	Sulfur dioxide	ppm	0-20	0.05
NO2	Nitrogen Dioxide	ppm	0-5	0.02
HCHO	Formamide	ppm	0-5	0.01
CL2	Chlorine	ppm	0-5	0.02
CO	Carbon monoxide	ppm	0-1000	0.5
H2S	Hydrogen sulfide	ppm	0-50	0.005
O3	Ozone	ppm	0-2	0.02

Table 7 ATMON FL measured substances

6.4.2. Driver development

ATMON FL driver was developed as a ROS node in Python3. Driver is deployed via roslaunch on robot platform as part of ROS there. Driver can be invoked with optional parameters: ``-p <device>`` to change default `/dev/tty` device that sensor registered to, and ``-s -f <file>`` to run simulation from data in specified file.

Driver itself is separated into few parts. SerialConnection object that manages serial parameters and reading from system device that sensor is registered to, DataPreprocessor that parses incoming messages and AtmonSensorDriverRosNode that publishes sensor data in ros messages.

SerialConnection and DataPreprocessor are running on dedicated listening thread to not cause delays with data receive process, then pre-processed data is put on thread-safe processing queue, from which AtmonSensorDriverRosNode that is running in main thread processes and sends the data.

Sending ROS messages starts when both types of packets are processed at least once. Driver does not store the data it receives. Below are the ROS message definitions for messages that are sent from the driver.

D4.3 Robots integrated into the system

AtmonSensor.msg 273 Bytes			Sensor.msg 392 Bytes		
1	uint64	timestamp	1	float64	value
2	float64	latitude	2	uint8	type # 101 PM2.5
3	float64	longitude	3		# 102 PM10
4	float64	altitude	4		# 2 Sulfur dioxide
5	uint8	RSSI	5		# 5 Nitrogen Dioxide
6	uint8	battery	6		# 13 Formamide
7	int8	status	7		# 7 Chlorine
8	Sensor	sensor1	8		# 3 Carbon monoxide
9	Sensor	sensor2	9		# 1 Hydrogen sulfide
10	Sensor	sensor3	10		# 8 Ozone
11	Sensor	sensor4	11		
12	Sensor	sensor5	12	uint8	unit # 1 ppm
13	Sensor	sensor6	13		# 5 µg/m3
14	float64	temperature	14	float64	range
15	float64	RH			
16	float64	pressure			

Figure 24 Atmon related ROS message schemas

6.4.3. Sensor simulator

Due to limited availability of ATMON FL, a sensor simulator was integrated into the driver. This simulator requires sample data to be provided in a standard text format with one packet per line. When a file with correct format is provided, the driver instead of reading data from device reads it from the file with a predefined frequency.

Due to one of defining features of Unix-like systems, which is that everything is a file, in this case makes this implementation fairly trivial – swapping input file (device) for input file (text). Additional encapsulation of this simulated serial connection made it completely transparent to developers, while allowing them to work on this software component even when physical device had to be returned to CNBOP.

6.5. MG-811CO2 sensor

Due to the limited availability of the ATMON FL, an additional sensor was integrated with the UGV, the MG-811 CO2 sensor. It can be used as a low-cost proof-of-concept gas sensor, when the main one is unavailable. The same sensor was also integrated with the UAV used in the ASSISTANCE project.

D4.3 Robots integrated into the system



Figure 25 MG-811 sensors, integrated with the UGV

MG-811 is an electrochemical sensor, without any embedded signal amplification circuits and with fairly specific requirements about power supply. To make the integration with the UGV faster, it was used in form of the development module, made by DFRobot. It contains the sensor with additional power converter and signal conditioning circuit, which makes its measurements less prone to external noise.

To integrate it with the UGV, it had to be made compliant with the common sensor interface. To do so and to make the possible future integration of other sensors easier, a simple development board was designed.

Custom enclosure was designed for the whole device, consisting of three main parts:

- base mounting plate, made of aluminum, to which MG-811 and RDB board are mounted,
- aluminum adapter for mounting the quick-release clamp for the NATO Accessory Rail, which is used to connect the whole device to the UGV,
- 3D-printed cover, made out of PET-G, mounted to the base plate using the press-fit inserts and screws,

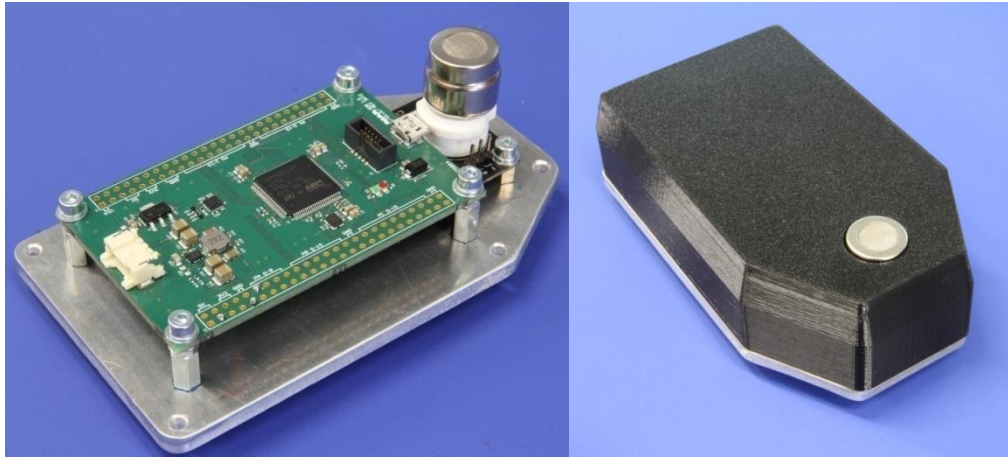


Figure 26 Base plate with mounted electronics and assembled device

6.5.1. Sensor calibration

According to the DFRobot, manufacturer of the MG-811 module, each sensor needs to be calibrated before use – it must be left for 48 hours in the place with clear air. The output voltage measured at the end of this process should be saved and used as a reference value for converting the sensor output from volts to the ppm (parts per million), which is commonly used unit in the CO₂ levels measurements.

To get some additional data about sensor calibration, three sensors were calibrated in the same time. Their output voltages were saved every 2 seconds through the whole process. To do so, a simple logging device was created, using the RDB 1.0 board. It is shown in Figure 27. Plots created from the data collected during this process are shown in Figure 28.

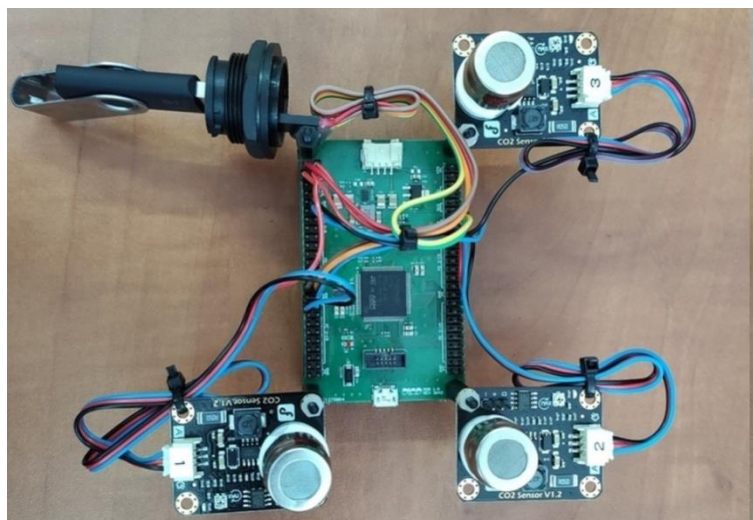


Figure 27 Calibration setup for multiple physical sensors

The MG-811 module manufacturer suggests using the measured clean air value and the value read from sensitivity curve from the sensor datasheet as a reference value for the volts to ppm conversion. However, the collected data clearly shows significant

D4.3 Robots integrated into the system

differences between sensors and their documentation. In order to increase the accuracy of the conversion from volts to ppm, additional measurement was made. The sensors were put in a sealed container with a concentration of CO₂ at about 10 000 ppm and left for about 10 minutes. Their output at the end of that process was used as a second reference value used in the conversion.

Both of the determined values are used for calculating the output of the MG811 – concentration of the CO₂ in the ppm.

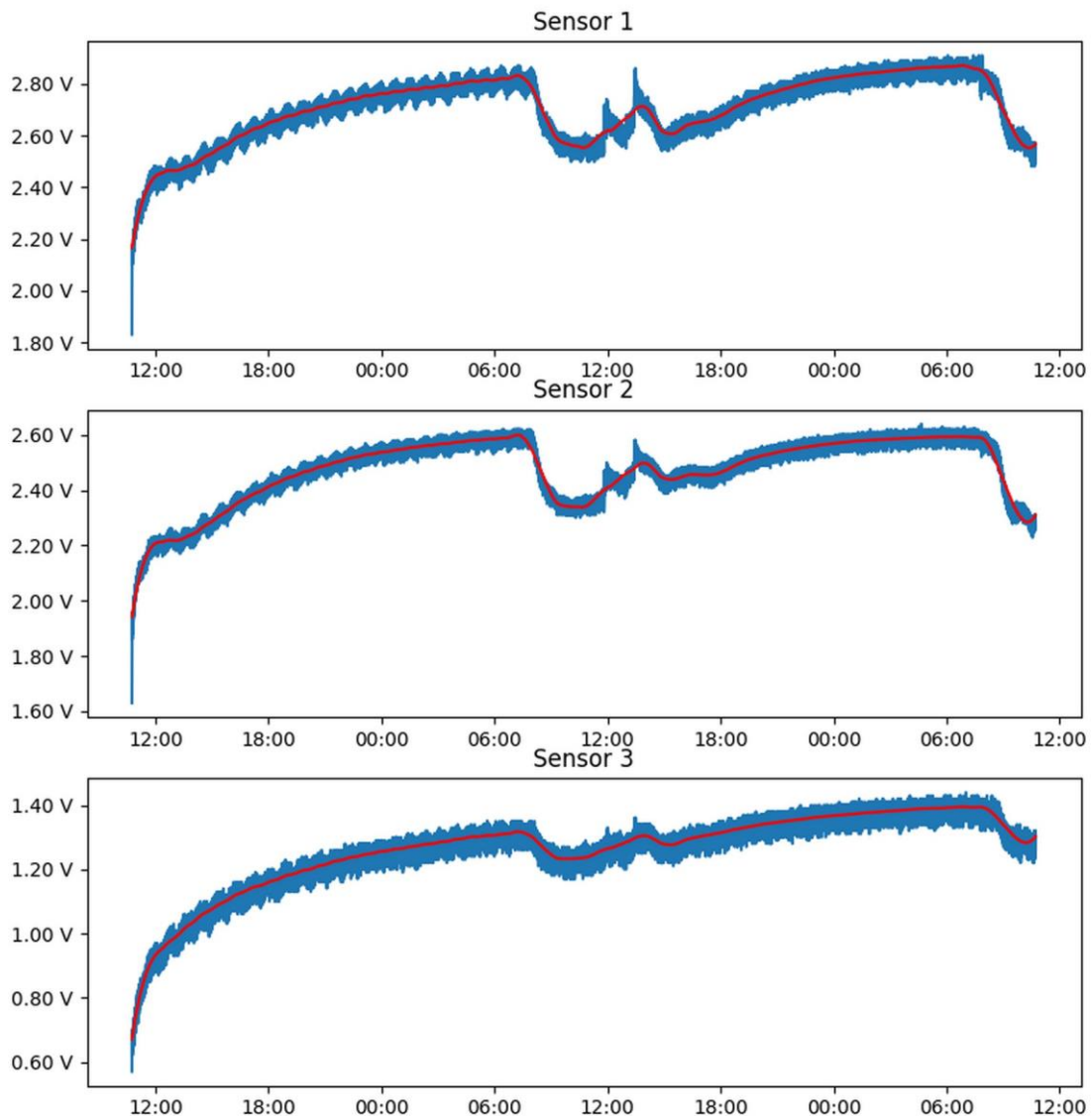


Figure 28 Data collected during the calibration process

6.5.2. Testing

The completed device was used on the UGV during multiple tests sessions. Output values are close to the ones outputted by the other sensors in stable conditions, however, characteristics of the electrochemical CO₂ concentration measurement

D4.3 Robots integrated into the system

causes two effects that have to be taken into account when analyzing data from this sensor:

- MG-811 needs to be sufficiently heated to output values comparable to other CO₂ sensors. Length of this process is highly dependent on the weather, it usually takes from 3 to 10 minutes until the output values stabilize.
- Due to the measurement method characteristics, it has a significant time constant – it takes from 5 to 30 seconds to get the stable output value when the CO₂ concentration level suddenly changes (for example, when the sensor is breathed on).

Despite these effects, the sensor fulfils its main task – it can be used as a low-cost proof-of-concept alternative, when the main gas sensor is unavailable.

6.6. ZR-2 radioactivity detector

6.6.1. Sensor description

Polon-Alfa ZR-2 radiometer module integrated in the scope of the FP7-SECURITY project EDEN was used. It was previously tested in a similar application and provides a real-time measurement of both dose and dose rates (dose accumulated per hour) for gamma radiation.



Figure 29 ZR-2 radioactivity sensor

6.6.2. Modifications

ZR-2 radiometer had to be modified to use the common sensor interface. The radio module was disabled and integrated batteries were removed, as both power and communication are now done through the same connector. It improved the sensor

reliability due to improved connection stability and eliminated the need of battery charging.

6.7. VAISALA WXT-520 weather station

6.7.1. Sensor description

Vaisala WXT520 weather station, also integrated in the scope of the FP7-SECURITY project EDEN was used. It was previously tested in a similar application and provides a simultaneous measurement of multiple parameters, such as:

- wind speed and direction
- air pressure, temperature and relative humidity
- rain and hail duration, intensity and accumulation

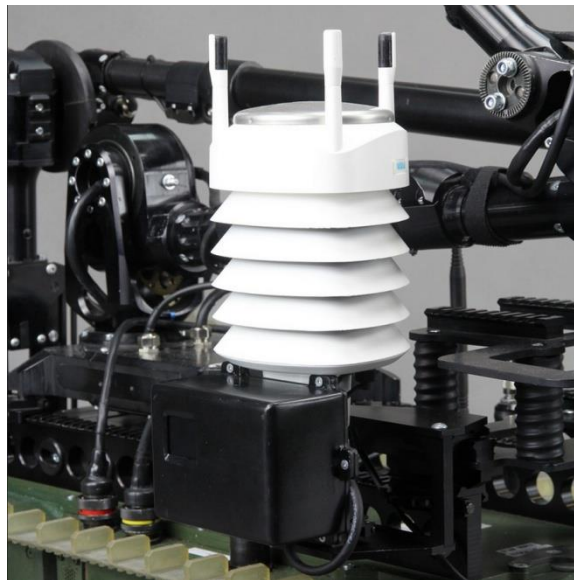


Figure 30 Vaisala WXT520 weather station

6.7.2. Modifications

Weather station had used the same interface as the ZR-2 sensor. To make it compliant with common sensor interface, it had to undergo the same modifications as described in 6.6.2.

6.8. EMF detector

Electric shock is a serious risk for a first responders. Potential hazards can be hard to identify without the use of the specialized equipment – downed live cables can be covered with debris, devices can be energized even after switching off the power due to improper wiring. End-users involved in the project has requested an ability of detection of dangerous voltages from the UGV, to fulfil that, an electromagnetic field sensor was designed and integrated with it.

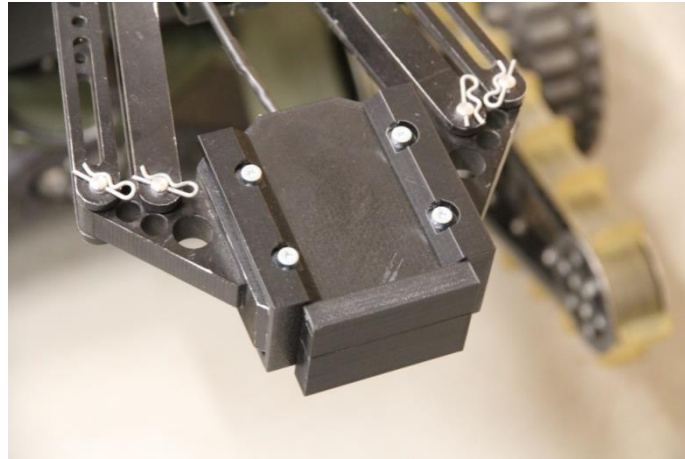


Figure 31 EMF sensor mounted in the UGV gripper

6.8.1. Market analysis

There are many commercially available non-contact voltage detectors dedicated for the emergency use. Most of them are handheld devices, often in a form of a wand, which increases the distance from the inspected object. Usually they don't provide any form of precise measurement of a electric field, but only indicate the intensity of it through the intensity of the sound or light signals. Example of such device is a AC HotStick™, shown on the Figure 32.

These devices are commonly used by the firefighters, however none of the available ones offer any type of data interface that could be used to parse their measurements. Any kind of integration with the UGV would require an extensive modification of the device and would not provide measurement in standard V/m units.

Another type of sensors capable of measuring electromagnetic field intensity are the multifield meters. These devices usually measure magnetic field, electric field and RF strength. Most of them do not have any kind of communication interfaces or they output only previously saved measurements, as they are intended for handheld use. Example of such device is a RS Pro Multi-field EMF meter, shown in Figure 33.



Figure 32 AC HotStick

D4.3 Robots integrated into the system



Figure 33 RS Pro Multi-field EMF meter

The last type of sensors capable of measuring the EMF are the specialized devices intended for the measuring the electromagnetic radiation of radio devices. These devices usually offer a communication interface, usually paired with an additional software for analysis of the measurement. However, due to their intended use, most of them are not capable of measuring the intensity of the electric field in the frequency band used in power grids.

Due to the limited availability of the COTS EMF sensors capable of being used on the UGV, a custom sensor was designed.

6.8.2. Hardware

Design of the custom sensor started with the research about the different methods of electromagnetic field measurement. After analysis and performing a series of tests on the breadboard, the D-dot sensor was selected – its operation principle is a charge induction between the grounded antenna and the energized device or cable, which can be used to measure the intensity of an electric field emitted by the source.

After a series of simulations, the final architecture (shown in Figure 34) of the measurement circuit was created and the design of the custom PCB had started, with certain requirements specified by the measurement process and the intended use of the final sensor:

- measurement circuit: compliant with the specified architecture, powered by separate, ultra-low noise symmetrical power supply,
- whole device should be galvanically isolated from the UGV – both power and communication interface, to separate the sensor from the external noise,
- communication interface: compliant with the common sensor interface

D4.3 Robots integrated into the system

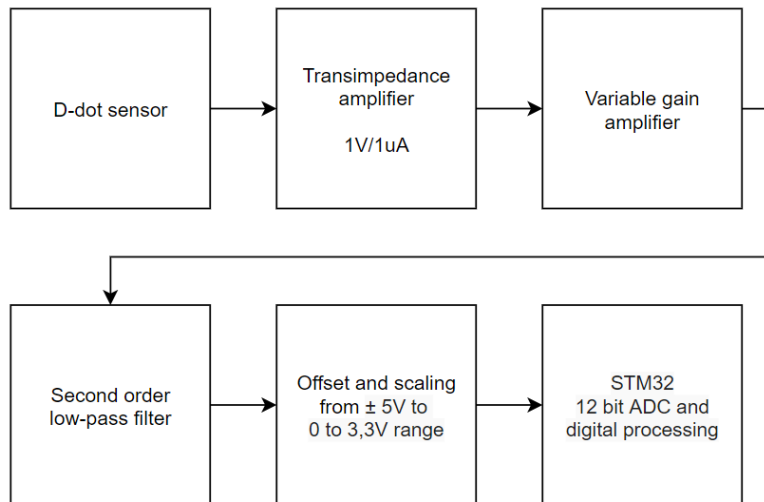


Figure 34 - EMF measurement circuit architecture

Final electronics is shown in Figure 35. It consists of the two boards, soldered together – the main one with all of the required components and the second one which is used as an antenna.

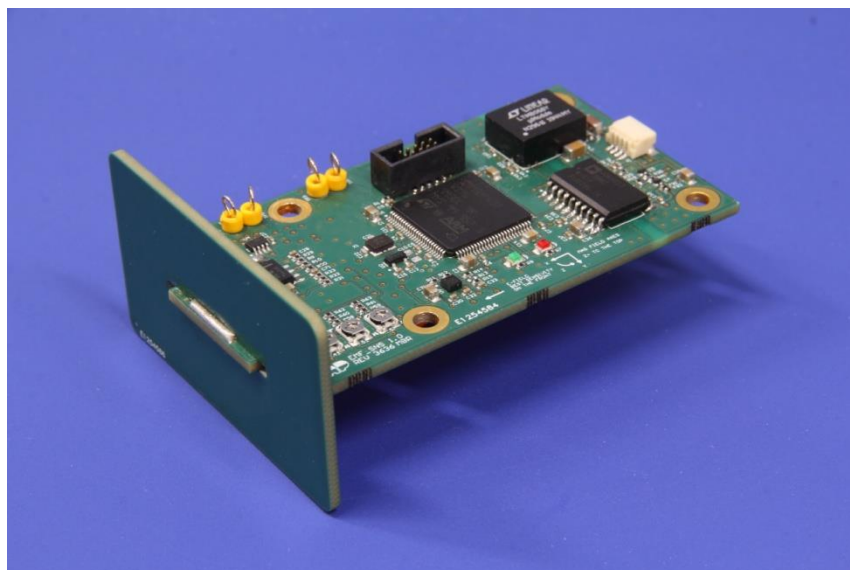


Figure 35 - EMF sensor electronics

Due to the characteristics of the sensor, its case must be made out of the non-conductive material, not to shield the antenna used for the measurement of the electromagnetic field. The 3D-printed enclosure was designed and used, as it made easier to test multiple variants of some of its elements, which is beneficial at the prototype stage. Final enclosure is shown in Figure 36.

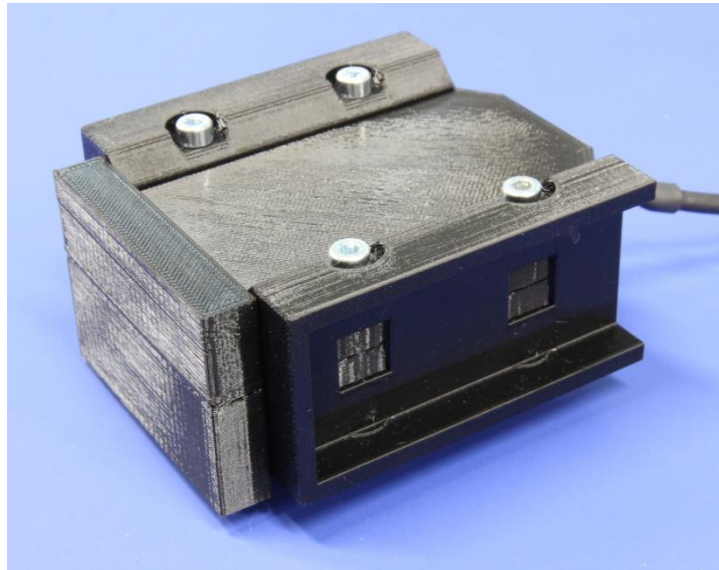


Figure 36 3D-printed EMF sensor enclosure

6.8.3. Firmware

Most of the devices using the D-dot sensors are usually using the fully analogue signal conditioning circuit, which amplifies, filters and integrates the input signal, resulting in an output voltage proportional to the electromagnetic field intensity. In the developed sensor, a different approach was used – output of the D-dot sensor is not integrated, FFT is used to determine energy carried by different frequencies.

FFT-based approach allows the sensor to measure electric field intensity in the frequency selected by the user – 50 Hz used in European power grids, 60 Hz used in the USA, lower frequencies used by some AC train system or 400 Hz used in some military aircrafts (compliant with MIL-STD-704 standard).

Firmware, that was created for the EMF sensor, implements all of the required digital signal processing and CANopen-based communication protocol.

6.8.4. Calibration

Initially, the sensor was outputting the EMF intensity – a value in range from 0 to 100% of a maximum EMF that can be measured on a selected gain setting. This approach allows the user to detect the EMF sources from the relatively far distance on the high gain settings and precisely locate them using the low gain one. However, this approach requires some experience in using the sensor to efficiently use this feature, as it can be unintuitive for new users and doesn't provide measurements in SI units.

To address this issue, an additional mode of operation was added to the sensor, in which it outputs the EMF value in V/m, which is the standard unit for measuring the strength of the electromagnetic field. Adding this feature required the determination of scaling factors for each gain setting.

D4.3 Robots integrated into the system

Measurements were done on the simple calibration setup, which consisted of a long, powered cable mounted on a table, that was used as an EMF source. On the table, lines were glued on, marking the distance from the cable: every 1 cm in the first 10 cm from the cable, and every 5 cm, in the next 40 cm. The whole setup was put in the room with low background electromagnetic field intensity.

To determine the scaling factors, the electric field intensity was measured multiple times on each of the lines and each of the gain settings. After that, the same measurements were done using an off-the-shelf sensor, RS PRO Multi-field EMF meter, which acted as a reference device.

Using the gathered data, the scaling factors were determined, which were used in the implementation of a new mode. The measurements from each of the gain setting, scaled to V/m using the determined factors, are shown in Figure 37.

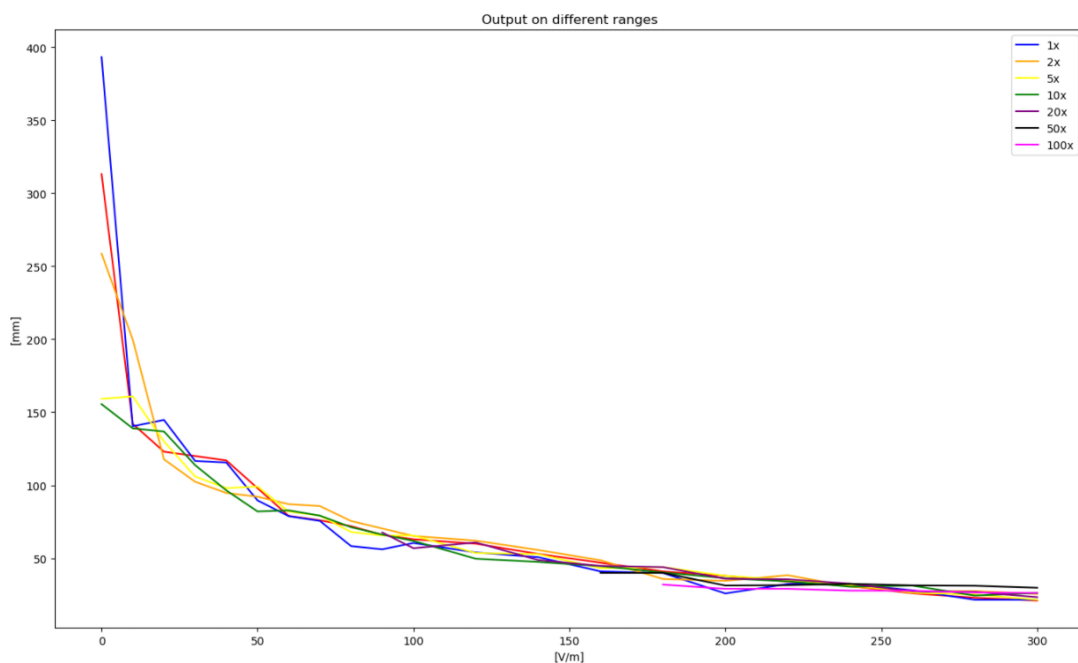


Figure 37 EMF output on different ranges after scaling

The sensor was tested during multiple test sessions; it meets all the requirements and performs as intended – allows the detection of powered devices from a safe range (relative to detected voltage) and outputs comparable values as other similar sensors.

6.9. CUTLANCA cutting extinguisher

6.9.1. Concept

CUTLANCA is a specialized tool designed and developed by and for firefighters. It sprays water at a very high pressure from small nozzle which allows water stream to cut through solid objects. It can cut through very sturdy materials such as concrete or steel. However, applying this stream requires the firefighter to stay very close to dangerous area, since the output nozzle has to be almost touching material that it needs to cut through. This in turn makes this an excellent application for UGV.

Rationale for inclusion of this effector is provided in deliverable D4.1 (Chapter 4.3.2). Device with all required additional hardware is being provided by ASSISTANCE project partner OSPOM.

Requirements that were defined for this solution on top of mechanical connector were of the usability nature. To be useful in real scenarios the robot needs to have about 300m effective range, which translates to capability to move with roughly 100m of 0.5 inch cable following with weight about 1 kg/m.

6.9.2. Mechanical integration

This tool in complete working form is comprised of multiple components – it requires a complete fire truck to work, however for integration with the robot only very simple mechanical interface had to be introduced. This interface was designed in a form of two mounting components shown in Figure 38.



Figure 38 CUTLANCA mechanical integration result

6.9.3. Testing

After initial integration concept tests were performed to validate the concept and determine requirements that would be required for proper implementation. The tests have validated platform rigidity, stress put on robot manipulator, possibilities for manipulator movement, effective range, ability to travel with water cable connected.

7. Summary

This document reports on the work performed in task T4.3 Robots Management and sensors integration that was required to achieve full integration with the ASSISTANCE system. All planned actions for task T4.3 have been completed successfully. Release of this document finalises the achievement of the project milestone MS4 “Unmanned platforms and control devices integrated”.

Following completion of task T4.3 next steps involve testing prototype in task T7.2 (M25-M35) based on validation plan submitted in D7.1 Validation Plan Report followed by demonstrator scenarios to finalize the project.