



— *ARSI Deliverable D26.7* —

MAV Platform Verification for Sewer Inspection Requirements



Table of contents

1.	Introduction and scope.....	3
1.1	Objective and scope	3
1.2	Structure of the document.....	3
1.3	References	3
1.4	Acronyms and abbreviations.....	3
2.	Laboratory tests	5
2.1	Autonomous control	6
2.2	Visual odometry	6
2.3	Altitude control.....	8
2.4	Takeoff and landing	9
2.5	Dynamic flight (wall following)	10
2.6	Costmap & local planning	11
3.	Tests in real sewers	13
3.1	Phase II evaluation area	13
3.2	Altitude control issues	16
3.3	Yaw control.....	17
3.4	Dust and small debris	18
3.5	Complex scenarios	20
3.6	Operational procedure	21
4.	Conclusion and future work.....	24
4.1	Lessons learned from Phase II.....	24
4.2	Goals for Phase III	24

1. Introduction and scope

1.1 Objective and scope

In this document we describe the verification process for the ARSI Micro-Air Vehicle (MAV) carried out throughout phase II of the ECHORD++ Sewer Inspection PDTI. Our development was guided by the requirements of the Challenge Brief [4] and the feedback from the phase I evaluation [7].

Our main objective in this phase was to develop a MAV prototype capable of navigating autonomously and safely in the sewer environments, thus demonstrating its usefulness as a tool for sewer inspection.

1.2 Structure of the document

The structure of this documents follows the main stages of the development and testing process in phase II:

- First, we concentrated on reviewing our platform design [2] in the light of the results of the phase I evaluation and feedback from the evaluators. Airframe and hardware changes are detailed in D26.6 – MAV prototype [5] and will not be duplicated here.
- Section 2 describes the various tests carried out in our laboratory to develop and validate the controlled flight and autonomy capabilities of the ARSI MAV, once functional prototypes for the onboard hardware and software were available.
- Having achieved a sufficient robustness in our laboratory, we started carrying out regular tests at the phase II evaluation area in Barcelona. We describe this process in section 3, in particular the main problems that we encountered and how we tackled them.

1.3 References

- [1] Challenge Brief – Robots for the inspection and clearance of sewer networks
- [2] D26.1 – Operation requirements and system design
- [3] D26.4 – Operational procedures and sewer inspection service
- [4] D26.5 – Prototype for sewer inspection
- [5] D26.6 – MAV prototype
- [6] D27.8 – Autonomous navigation and data recording
- [7] Sewer – Final evaluation phase I
- [8] [TeraRanger One performance over water](#)

1.4 Acronyms and abbreviations

- ARSI: Aerial Robot for Sewer Inspection
- LoS: Line-of-Sight
- MAV: Micro Air Vehicle
- [Mavlink](#): MAV communication protocol

- [Mavros](#): Mavlink-ROS interface
- MoCap: Motion Capture system
- PCL: [Point Cloud Library](#)
- PID: Proportional-integral-Derivative controller
- PX4: Pixhawk flight stack
- RGB-D: RGB and Depth camera
- ROS: Robotic Operating System
- [RVIZ](#): ROS visualization interface
- ToF: Time-of-Flight

2. Laboratory tests

In this section we describe the laboratory tests executed in the flying arena at Eurecat. The test area is shown in Figure 1 consists of a 8m x 6m rectangular room with 4m high ceilings and protective netting.



Figure 1: ARSI flying arena at Eurecat

Table 1 lists the various MAV functions developed and tested over a period stretching from the start of phase II in November 2016 to July 2017, after which all tests were carried out in real sewer networks (see section 3). A mid-phase presentation and demonstration was carried out in March. The MAV functions are described in the following subsections, along with corresponding tests and results.

Functions	Nov	Dec	Jan	Feb	Mar	Apr	Jun	July
Autonomous control								
Visual odometry								
Altitude control								
Takeoff and landing								
Dynamic flight (wall following)								
Costmap & local planning								
Complex scenarios								

Table 1: Laboratory tests schedule throughout phase II

2.1 Autonomous control

The Pixhawk autopilot is designed to allow MAVs to fly autonomously. Using the Mavlink ([Micro-Air Vehicle Communication Protocol](#)) users can issue control requests to the Pixhawk as position or velocity “setpoints”. Knowledge of the MAV position and velocity states is required to achieve this setpoints, in order to calculate errors that the control loops then correct for. Since the typical use of MAVs is outdoors, the Pixhawk’s localization system relies on GPS as its primary position source, aided with various onboard sensors including a barometer, an IMU, and a magnetometer. However GPS is not available in underground sewer networks, therefore the ARSI system requires a replacement positional source for autonomous flight.

In this first phase of our development we used an [OptiTrack](#) Motion Capture (MoCap) system as our position source. This system relies on a set of calibrated infrared cameras to detect special markers mounted on the object to track. We mounted 8 cameras on tripods and placed them around our flying arena, before calibrating them using the OptiTrack software to calculate their relative poses with high-accuracy. By placing the markers on the ARSI MAV, we were able to use OptiTrack and receive high-accuracy measurements of its position and velocity states, which were then transmitted to the Pixhawk localization module over Mavlink.

The Pixhawk autopilot uses 3 cascaded PID controllers (for position, velocity and attitude) which must be tuned specifically for the dynamics of the ARSI MAV to achieve stable control during autonomous flights. Tuning a MAV controller is notoriously difficult due to the number of degrees of freedom and layers in the control system. Moreover, state estimation impacts control independently of how well tuned it is, since it affects error calculation (eg. the position error to a given setpoint).

By using OptiTrack to obtain a near-perfect estimation of the MAV position and velocity, we were able to isolate control errors whilst executing simple setpoint patterns and perform a robust calibration of the PID gains for each control loop. A video of the tuning process using OptiTrack is available [here](#) (the cameras are mounted on black tripods).

2.2 Visual odometry

Once the Pixhawk control loops were calibrated, we integrated the [Orbbec Astra](#) RGB-D camera and the visual odometry algorithm [RtabMap](#) on the ARSI MAV. The camera driver and visual odometry are both ROS nodelets running on Intel NUC i7 onboard computer which replaced the Odroid PX4 used in phase I due to CPU and memory requirements. RtabMap generates real-time estimates of the vehicle pose and velocity at a rate of ~15Hz, which are then transmitted to Pixhawk over Mavlink using the Mavros vision_pose_estimate plugin.

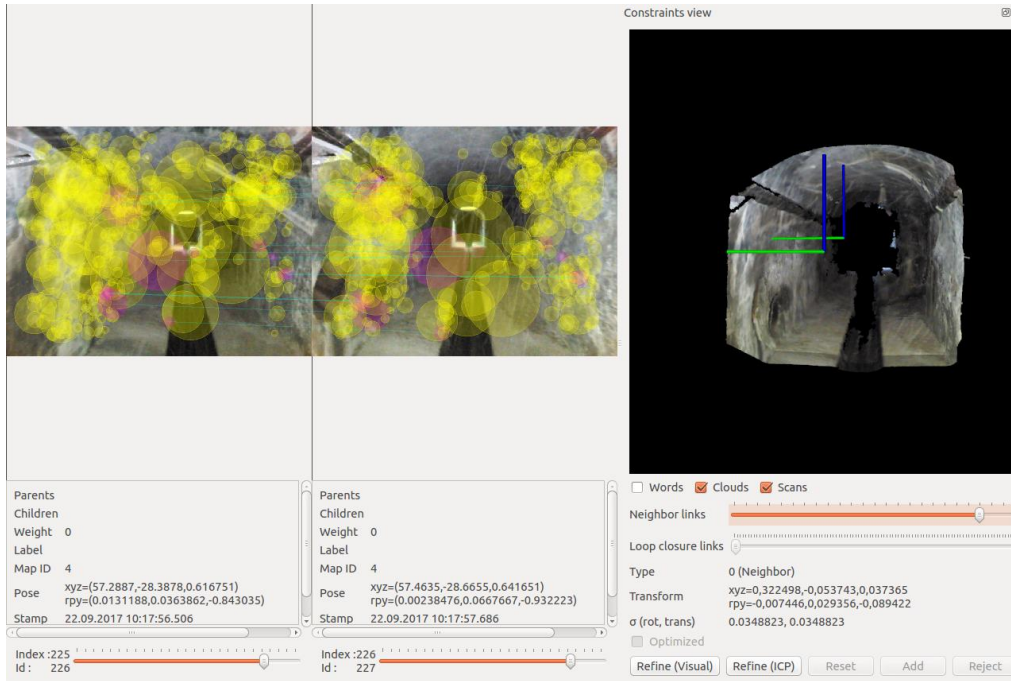


Figure 2: Detection and matching of visual features using RtabMap

The detection and matching of visual features across successive frames using RtabMap is illustrated in Figure 2. Figure 3 depicts a trajectory (in red) calculated by RtabMap in our flying arena. The justification for choosing a RGB-D camera and RtabMap as well as the integration process are given in deliverables D26.5 – Prototype for sewer inspection [4] and D26.6 – MAV prototype [5].

We tested this configuration by executing pre-programmed setpoint patterns in our flying arena, and estimated the RtabMap trajectory to be within 20cm of the MoCap ground truth over distances of 10 to 15m. However as discussed in D26.6 – MAV prototype [5], drift over time in the visual odometry trajectory is to be expected, especially in the sewers where features are scarce and loop-closing is rarely possible. While this drift can be corrected for in post-processing (see details in D27.8 – Autonomous navigation and data recording [6] and this [video](#) of map building using RtabMap) it does not prevent the ARSI MAV from navigating autonomously in the sewers since our strategy relies on dynamic planning, where local trajectories are recalculated in real-time using data from the onboard sensors (see details in following sections).

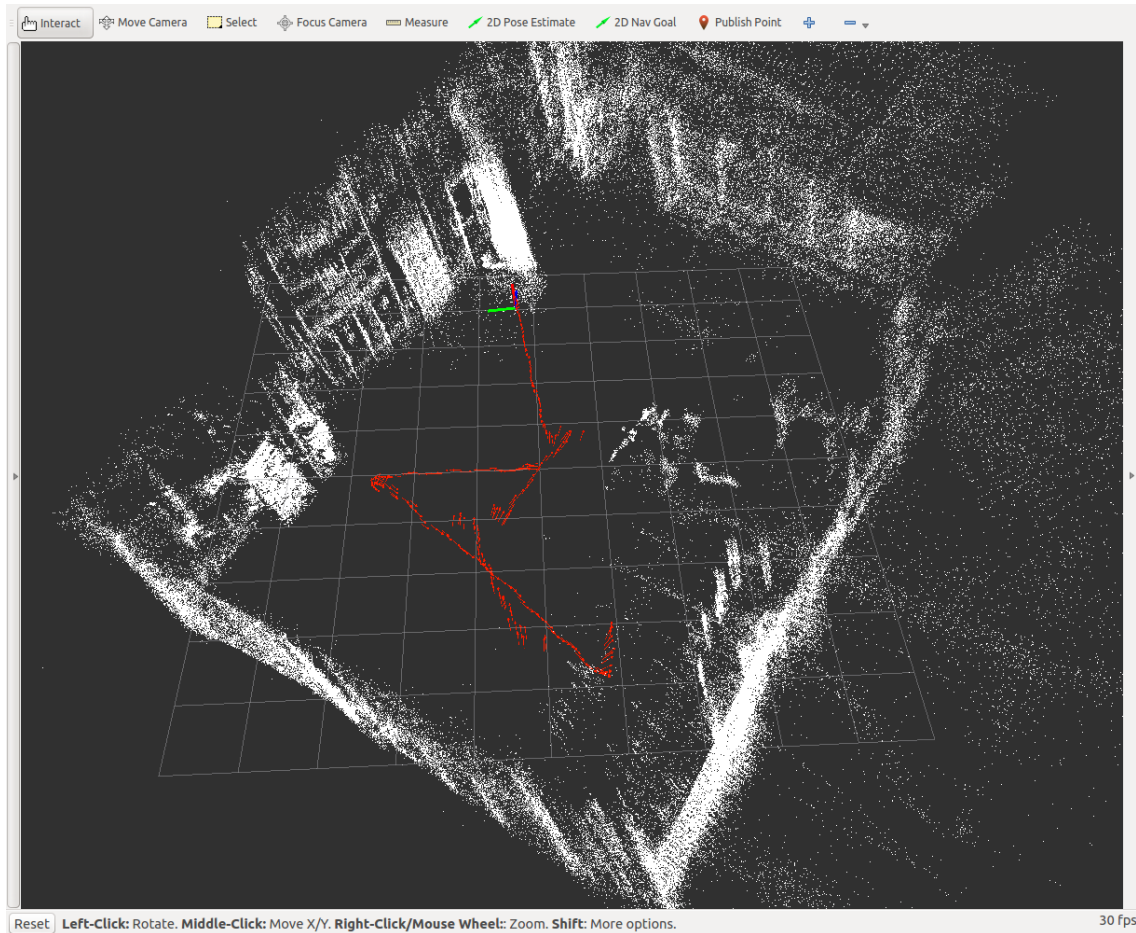


Figure 3: MAV trajectory (in red) estimated using RtabMap in our flying arena

2.3 Altitude control

As detailed in D26.6 – MAV prototype [5], our system requires a dedicated altitude sensor. In phase II we integrated a Time-of-Flight (ToF) infrared sensor as our primary sensor for altitude control. We initially worked with the [TeraRanger One](#) and then with the [TeraRanger Multiflex](#). Both sensors are infrared ToF rangefinders, however their specifications differ significantly as shown in Table 2.

	<i>TeraRanger One</i>	<i>TeraRanger Multiflex</i>
<i>Range</i>	0.2 – 14m	0-2m
<i>Frequency</i>	1KHz	30Hz divided by # of sensors
<i>Weight</i>	8g + USB hub 20g total	1 sensor: 2g 8 sensors + hub: 20g

Table 2: TeraRanger sensors specifications

The TeraRanger Multiflex was very interesting to us for two reasons: 1) it is one of the lightest sensors on the market (2g per sensor) 2) it allows connecting up to 8 sensors to a single hub and

create custom configurations using flexible cables (see Figure 4). This is particularly useful for ARSI as one sensor can be mounted underneath the MAV to generate ground range measurements, while the rest could be mounted in a circular pattern to generate laserscan-like measurements, thus replacing the 2D laser and significantly reducing the overall payload weight.

However our laboratory tests showed that the number of sensors and the reduced update frequency (30Hz divided by the number of sensors = 4Hz) were not adapted for real-time navigation in an environment as narrow as the sewer tunnels, where fast updates and a wide, continuous field-of-view are required in order to achieve safe flight (see also details in section 3.1). We hope to work with the TeraRanger manufacturers in phase III of this project to see whether these shortcomings can be overcome, which would allow one or more Multiflex sensor strips to be integrated on the ARSI platform.



Figure 4: Example configurations of the TeraRanger Multiflex

Both TeraRanger sensors are easily integrated using open-source ROS drivers and the Mavros distance sensor plugin, which transmits range data to the Pixhawk autopilot using the Mavlink protocol as a [MAV_DISTANCE_SENSOR](#) message. Pixhawk uses these unbiased range measurements to correct for acceleration errors along the Z-axis in its estimation of the vehicle pose.

Pixhawk implements a simple strategy to cope with uneven or irregular grounds below the MAV: a ground plane defined by the initial TeraRanger measurements is updated if successive new measurements exhibit a range difference above a user-defined threshold. We tested this functionality by flying the ARSI MAV over boxes, to simulate a temporary change in ground altitude. As the ARSI MAV flies over the box, Pixhawk detects that the incoming ranges have changed. The ground plane is updated as well as the altitude request relative to the ground, and the MAV keeps flying at the same altitude as it passes over the box.

2.4 Takeoff and landing

The Pixhawk flight stack implements an “offboard” mode in which users can take control of the MAV by issuing control requests in the form of position or velocity setpoints. As a safety measure, setpoints must be issued continuously and at a rate greater than 2Hz, or Pixhawk will consider that the offboard link was lost and will enter a failsafe mode where an emergency landing is executed.

As detailed in D26.6 – MAV prototype [5] section 3.3, sewer inspections using the ARSI MAV are executed as a series of configurable goals such as “takeoff”, “follow path”, “land”, “disarm”, etc. Each goal is internally executed as a series of position or velocity setpoints, but this is abstracted out in our Mission Execution software so that ARSI users are only exposed to general MAV concepts regardless of the autopilot backend.

In this phase we developed autonomous takeoff and landing goals:

- Takeoff is executed as a position setpoint positioned at the selected altitude (in body frame) over the initial MAV position.
- Landing is executed as a series of velocity setpoints generating a controlled downward movement towards the ground. When the ground is detected using velocity and ground range thresholds, the MAV is considered “landed” and the motors are stopped (“disarmed”).

Both goals were developed in parallel to the TeraRanger integration, since they require on an accurate altitude estimation. They were tested extensively in our test arena, and throughout phase II since they form part of every mission (see for example this [video](#)).

2.5 Dynamic flight (wall following)

Our next task was to develop a first dynamic flight mode, in preparation for the demonstration with UPC evaluators in March. Our task was to demonstrate fully autonomous flight and the ability to navigate a complex environment using only the MAV onboard sensors.

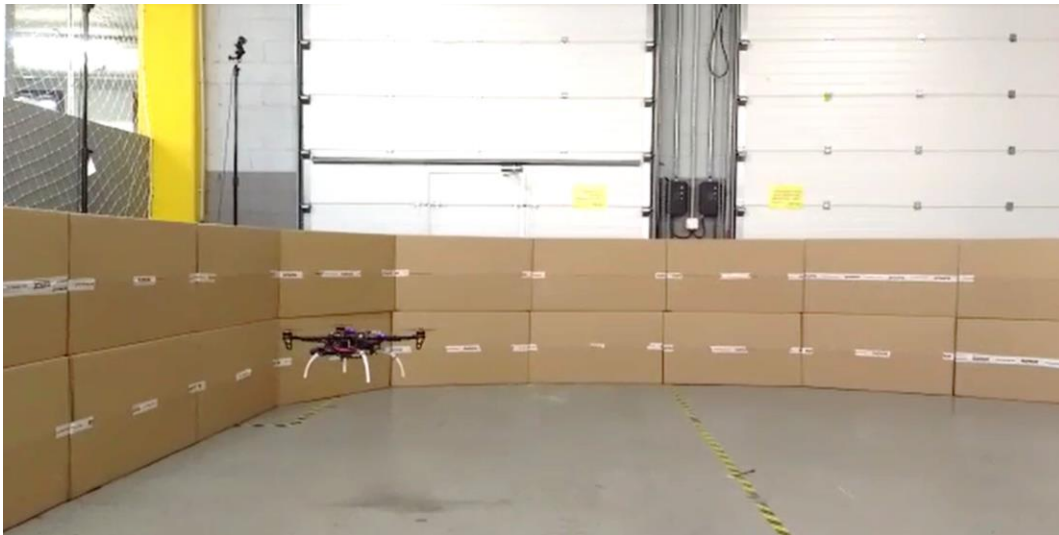


Figure 5: Cardboard boxes setup for wall following

At this point we started using cardboard boxes to simulate various configurations of walls, tunnels and intersections (see for example Figure 5). These boxes are visible in the RGB-D point clouds for obstacle detection, and they exhibit enough visual features (in particular the seams and strips of duct tape used to keep them closed) to produce robust visual odometry.

Figure 6 depicts the wall following process. Using the open-source [Point Cloud Library](#) (PCL) we estimate surface normals and segment RGB-D point clouds to extract the vertical walls (in white). Then an exclusion zone is created around the wall based on a configurable range. Setpoints are then generated along the contour, facing a given direction so that the MAV sees far enough ahead to plan for sharp turns. The algorithm processes point clouds at 20Hz and constantly generates new setpoints which are then issued to the Pixhawk autopilot via Mavros and Mavlink.

A video demonstrating autonomous wall following is given [here](#).

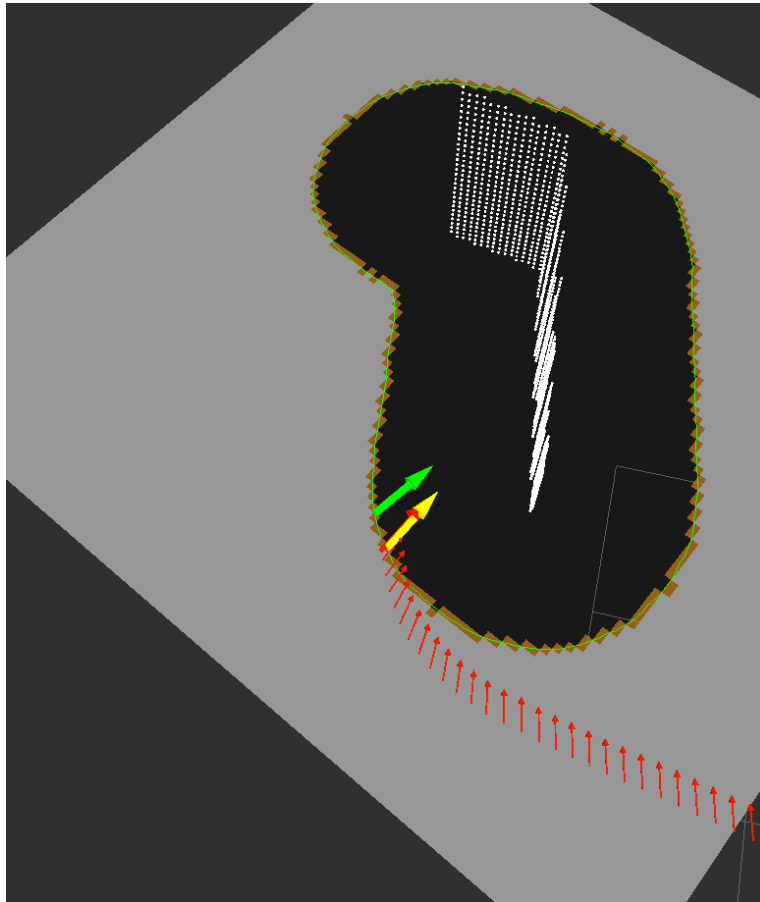


Figure 6: Wall following using RGBD point clouds

2.6 Costmap & local planning

Our next task was to develop the core of our navigation system for sewer networks. As already detailed in D26.6 – MAV prototype [5] our approach follows the global planner / costmap / local planner architecture commonly used in robotics.

Our system does not require a global planner since sewer inspections are planned in advance by the operators based on GIS information (typically entry and exit manholes). Our costmap module is an extension of the ROS [costmap_2d](#) with some additional features (in particular the dust filter) while our local planner is derived from the ROS [dwa_local_planner](#) (see [5] for design details on

both modules).

Figure 7 depicts the costmap and local planner during a flight in Mercado del Borne. Laser data (in red) is projected onto the 2D costmap and used by the local planner to calculate an optimal trajectory (in dark blue) to follow a global path (in pink). At each planner iteration, a setpoint (light blue) is derived from the trajectory and sent to the Pixhawk autopilot, so that the estimated MAV position (in green) follows a safe path towards the goal.

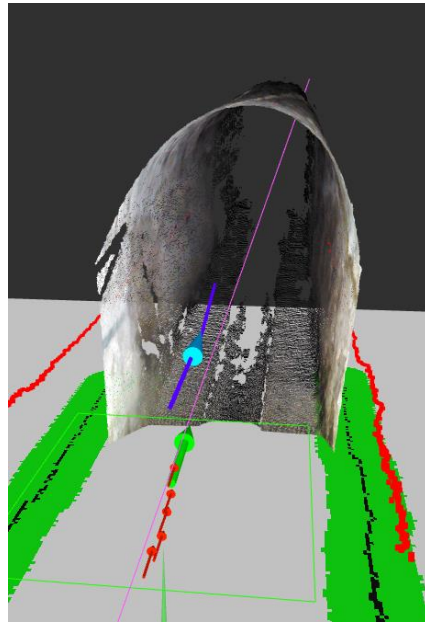


Figure 7: Costmap (in green) and trajectory (in blue) generated by the ARSI local planner

In the following months, we used the cardboard boxes to simulate a variety of sewer-like scenarios, including tunnels of various widths (down to ~70cm), intersections, turns (see for example Figure 1), obstacles, etc. Countless test flights were carried out, allowing us to develop and troubleshoot our MAV navigation system as well as the onboard sensors in a controlled and safe environment. These tests are detailed in the progress report D26.5 – Prototype for sewer inspection [4] and a summary video is available [here](#). Concerned that the short length of our test flights could conceal longer-range control issues, we also conducted several ~40m flights in our parking ([video](#)).

By the end of this phase, our laboratory tests had shown that the ARSI MAV was able to navigate safely and robustly in what we considered to be sewer-like environments. In the next section we describe our subsequent tests in real sewers at Mercado del Borne in Barcelona, in preparation for the phase II evaluation.

3. Tests in real sewers

Based on the results of the laboratory tests, in July we decided that the platform and the onboard software was stable enough to attempt flights in the real sewers. Between July and October, we carried out a total of 17 days of tests with the ARSI MAV at the location of the phase II evaluation in Barcelona (see test calendar in Figure 8). In all our tests, access to the sewers and logistical support was provided by ARSI partner FCC. In addition to these flight tests we visited the sewers several times in phase II for data collection as well as communications tests.

July						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

August						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

September						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

October						
Mon	Tue	Wed	Thu	Fri	Sat	Sun
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Figure 8: Calendar of ARSI MAV tests in Mercado del Borne before the phase II evaluation (October 16-18). October 24-25 were long-range parking tests

Since our consortium is based in Barcelona, we initially conducted single-day tests so that issues encountered (control issues in particular) could be investigated thoroughly back at base, and so that any control tuning or onboard software changes could be validated in our test area at Eurecat before returning to the sewers.

In the following sections we describe the various tests carried out in the sewers in relation to the functions that they validated. If problems were found, we explain how they were resolved.

3.1 Phase II evaluation area

First we will briefly describe the evaluation area chosen for phase II of the ECHORD++ Sewer Inspection PDTI, in particular its most complex features, as a reference for the following sections.

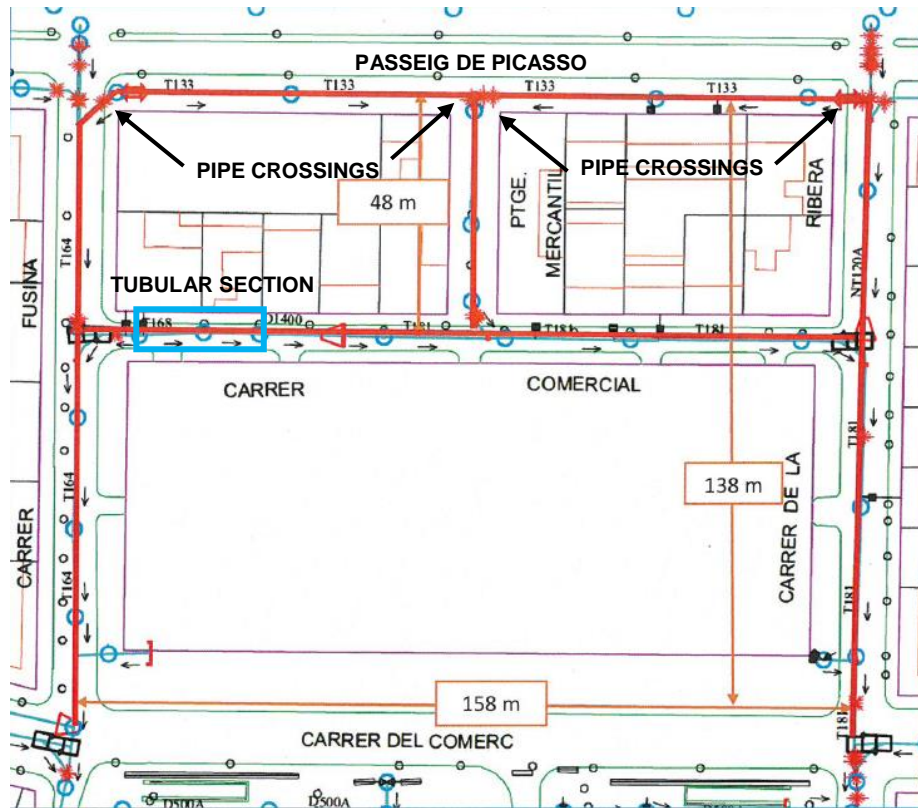


Figure 9: Phase II evaluation area in Barcelona

The phase II evaluation area is defined by the streets marked in red in Figure 9:

- Calles (streets) Fusina, Ribera and Comercial are ~140m-long straight lines of 120cm wide tunnels, apart from the narrower top section of Ribera (80cm wide). Their heights vary slightly around 170cm.
- Calle Comercial features a ~25m tubular section with a diameter of 140cm (marked on the map in blue). This section is made of plastic material and is challenging due to the lack of visual features (see Figure 11). Also there is no central drain, and thus no dry area for the MAV to land in case of an emergency.
- Passeig de Picasso is a narrow tunnel where large service pipes (marked as red stars on the map) intersect the tunnel at each intersection, leaving only 60cm of clearance for the MAV (see Figure 10). All these pipes are situated in narrow (<1m) turns.
- All sewer tunnels feature lateral inlets allow sewage water be evacuated from neighboring buildings (see Figure 12). While most inlets lie near the ground to flow directly in the central drain, some can be situated at mid-height or above.



Figure 10: Service pipe in Passeig de Picasso



Figure 11: Tubular section in Calle Comercial



Figure 12: Water flow from an inlet in Calle Ribera

3.2 Altitude control issues

From the very first tests in the sewers, we were faced with instability in altitude control. We observed constant 30 to 40cm oscillations along the Z-axis during flights, and even several crashes against the roof of the tunnels when control was lost entirely.

In this section we describe how this issue was investigated and how we were able to bring the altitude control accuracy to a level good enough to ensure flight safety as well as quality inspection data.

TeraRanger range sensor

As explained in D26.6 – MAV prototype [5], a dedicated range sensor is required to accurately control the altitude of the ARSI MAV during flight. As previously discussed, we experimented the [TeraRanger Multiflex](#), which had shown good results in our laboratory even though we had seen that its operational range was significantly less than the 2m advertised (~1m in reality). In the sewers, range measurements were repeatedly lost even at low altitudes, perhaps due to the presence of sewage water in the central canal. Quickly we realized that the sensor in its current state was not adapted to our MAV in sewer conditions, and reverted to using the TeraRanger One and its proven higher-end characteristics (14m range, 1KHz refresh rate).

Noisy range measurements

As presented in [8], the TeraRanger One was tested with good results for operations over water, as is often the case in the sewers since the MAV often flies directly above the central sewage drain. However, we also confirmed the study's that range measurement over water is significantly noisier (Figure 13). Since this noise would affect velocity estimations and control, we introduced a simple low-pass filter to reduce range measurement noise.

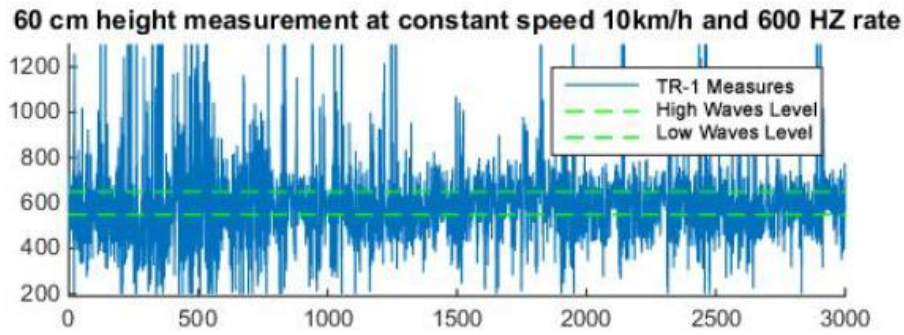


Figure 13: TeraRanger measurement noise over water

Altitude control loops tuning

Altitude control had to be re-tuned for the conditions of the sewers, in particular to avoid the type of oscillations shown in Figure 14 and the MAV colliding with the sewer roof. We chose a low p value for the outer loop to minimize oscillations, while the inner loop is more reactive, and obtained good control as shown in this [video](#). Our analysis of the control responses suggests that control can be further improved if the payload weight is reduced in phase III, as discussed in section 4.

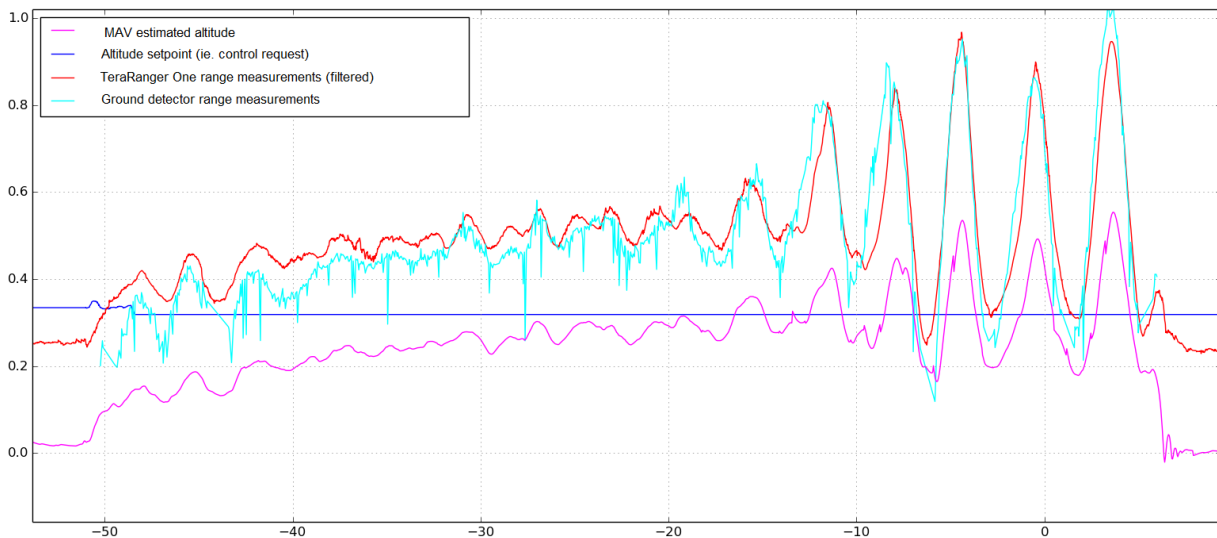


Figure 14: Altitude oscillations resulting in the ARSI MAV colliding with the sewer roof

3.3 Yaw control

Yaw control is particularly critical for a number of reasons:

- Sewer tunnels are very narrow (at most 120cm wide in Mercado del Borne, down to 75cm), therefore there is little margin on each side of the MAV during flights. Yaw error increases the lateral footprint of the drone and increases the chance of impact with the walls.
- Rapid changes in yaw cause blur in the images from the RGB-D camera, which is

detrimental to feature detection and therefore to the quality of the visual odometry solution. This in turn results in poor velocity estimation, reducing the capacity of the Pixhawk control loops to reduce yaw error.

- In a narrow sewer, yaw error implies that the MAV is facing towards a wall instead of the direction of the tunnel. Given the short distances, the direct illumination from the MAV LEDs can saturate the images, which is also detrimental to visual odometry and velocity estimation.

For these reasons, large errors in yaw are difficult to recover from in narrow sewer tunnels, and have caused numerous forced landing or crashes throughout our tests. Yaw control had to be re-tuned for the conditions of the sewers. The key was to ensure that the inner loops (yaw rate in particular) were reactive enough by raising both the proportional gains and request limits. Stable yaw control along a sewer tunnel in the phase II evaluation area is shown in this [video](#).

3.4 *Dust and small debris*

During our tests we noticed accumulations of dust and other small debris in some sewer tunnels. While we did not anticipate this issue based on the Challenge Brief or our visits to the phase I location, these particles are lifted and agitated by the air flow of the MAV motors. In parts of the phase II evaluation area, the concentration of particles in the air was dense enough to be visible in the imagery of the frontal camera as well as in the laser data.

In particularly narrow and confined areas, the concentration of particles in the air can get so dense that it dramatically affects visibility in the frontal camera. Figure 15 shows an extreme case of this phenomenon in the phase II evaluation area in Barcelona. Dust is notably an issue when flying very close to the ground, as is required to overcome obstacles such as service pipes in Passeig the Picasso (see map in Figure 9).

In terms of the video imagery, we mitigated this problem by adjusting our LED configuration so that we would avoid illuminating dust particles directly in front of the camera. We also adjusted parameters in RtabMap to minimize the chance of dust particles being considered as visual features. Overall we found that RtabMap was sufficiently robust to the issue, due to the fact that dusts is typically only present for short periods of time so that subsequent frames contain enough stable features for the algorithm to resume visual tracking.



Figure 15: Extreme case of imagery contaminated by dust lifted by the MAV motors

Dust particles can also become visible in laser data, and are consequently added to the costmap and treated as obstacles by the local planner. Since they are typically unstable and rapidly moving, they cause the local planner to incorrectly adjust its trajectory and change control requests at each iteration, which in our tests typically resulted in loss of control of the ARSI MAV.

We addressed this problem by introducing a “dust filter” in our laser data processing pipeline. We used a radius-based outlier filter, where data points are discarded if the number of neighboring points within a certain radius is too low. In Figure 16 for example, the yellow and green samples could be discarded for having little or no support, while the blue sample would be retained.

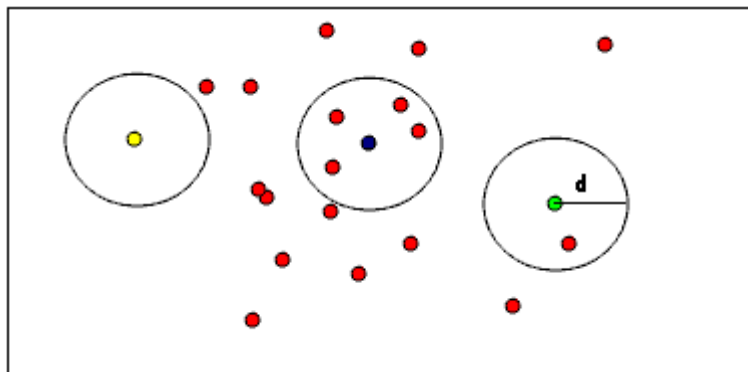
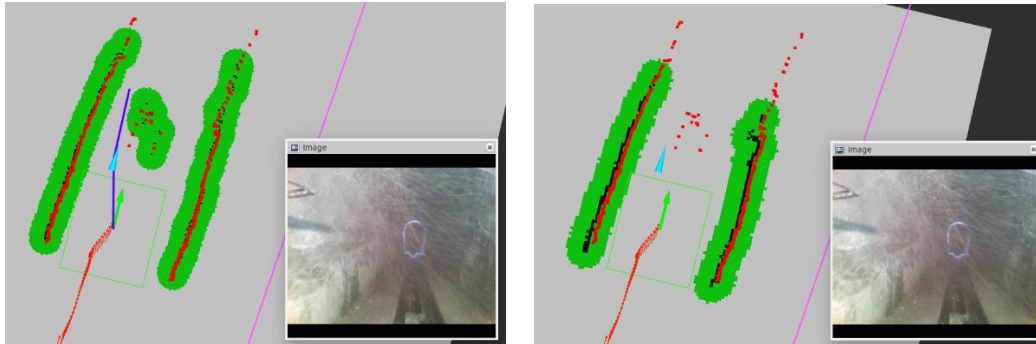


Figure 16: Radius-based outlier removal (image courtesy of PCL)

Since laser scans have fixed angular intervals, spacing between samples increases with their range to the sensor, and so the filter radius must be proportionally increased when processing laser data further ahead. This is particularly important when the MAV navigates intersections or similar complex areas, where loss of longer-range laser data results in the local planner reacting

to changes in the sewer configuration at the very last moment. This typically results in very aggressive control requests likely to destabilize the MAV. By adjusting the radius based on range, dust particles can be filtered out effectively without reducing the prevision range of the costmap and the local planner.



*Figure 17: (left) Local planner avoiding dust particles flagged as obstacles
(right) Costmap after radius-based dust filtering*

3.5 Complex scenarios

Once the altitude and yaw control became stable, and dust particles were largely removed from the costmap, we were able to carry out longer flights as well as focus on the complex parts in the phase II evaluation area.

In terms of the local planner, the most complex scenarios are intersections and connections between sewers of different shapes and sizes. In these situations the key parameter was the prediction distance, which defines how far ahead from the MAV location the planner should look to plan the next trajectory towards the global goal. As previously discussed, if the distance is too short the MAV will react at the very last moment to changes in the costmap, likely causing aggressive movements which can destabilize the MAV. If the prediction distance is too large, the planner will focus on more distant areas of the costmap, and it will generate coarser and potentially less safe trajectories in narrow turns or intersections.

Our tests showed that a predication range of 1m produced good stability in the intersections and turns in the Mercado del Borne area, and we intend to improve our planning algorithm in phase III to produce safe and efficient trajectories in all situations. Flights by the ARSI MAV in complex areas are shown in the following videos (note that the pilot is for emergency only, and does not control the MAV):

- [Flight through a narrow intersection with turn](#)
- [Low-altitude flight underneath a service pipe](#)



Figure 18: ARSI MAV flying avoiding a service pipe

3.6 Operational procedure

As flights with the ARSI MAV became more stable, we started to move away from our “engineering” test setup to start following the ARSI operational procedures presented in D26.4 – Operational procedures and sewer inspection service [3]. Early tests were conducted with an engineer and a safety pilot both inside the sewers, controlling and monitoring the ARSI MAV. We moved to executing missions from the surface, connecting to the Wi-Fi router over Ethernet and using our Mission Control interface (see Figure 19 and detailed description in [6]).

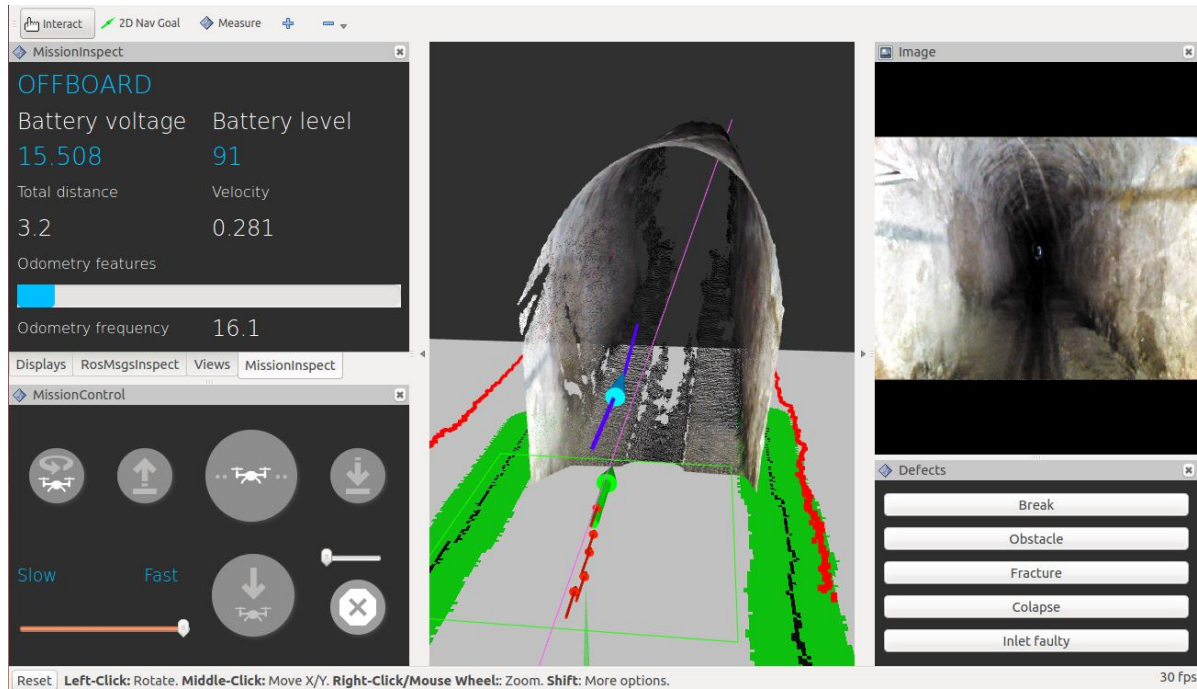


Figure 19: ARSI Mission Control interface

We also prepared a rugged watertight case for the Wi-Fi router, so that it could be deployed and recovered directly from the surface using ropes (see Figure 20), reducing the number of times inspection personnel was required to enter the sewers.

ARSI partners worked together so that FCC's inspection teams could have a hands-on experience of the ARSI system. With minimal training, they were able to deploy and recover the MAV in the sewers, replace batteries, and start the system ready for flight. Through this process we received their feedback not only on the MAV itself but also on the operational procedures.



Figure 20: Rugged and watertight case for Wi-Fi router

4. Conclusion and future work

4.1 *Lessons learned from Phase II*

As described in Section 3, our flight tests in real sewers immediately highlighted serious control issues, notably instability in altitude and yaw. The margins of error in sewer environments are so narrow that these control problems often resulted in forced landings or even crashes, with a risk of damaging the airframe or the expensive onboard sensors. Testing in the sewers is even more difficult because of the presence of water, since crashes are likely to lead to water damage in the MAV electronics.

The process of identifying, analyzing and correcting these issues proved much more time-consuming than anticipated. Interestingly, none of these control issues occurred in our test environment, even as we tried to reproduce as many of the adverse conditions present in the sewers as possible including reducing visual features, working in total obscurity, simulating a sewer drain, or even throwing dust at the MAV during flights. Our educated guess is that the turbulences generated by the MAV itself when flying in confined sewer tunnels pushed our control system to its limits and highlighted shortcomings, in the control loops in particular, that did not manifest themselves in our laboratory.

Our strategy in this phase was to demonstrate the robustness of the system in laboratory tests, primarily to reduce the risk of damage to the MAV or the payload when testing in the real sewer. However the first real sewer tests revealed critical control issues that were not shown during laboratory tests and impeded our planned continuous progress towards the evaluation. A lesson learned from that Phase II with respect to the test plan is that the MAV solution must be tested as early as possible in the real environment it is intended for. For instance, manual flights could have flagged the dust problem, and short autonomous flights could have perhaps highlighted some of the control issues.

We hope to take this experience with us into phase III in order to work more efficiently and achieve the robustness required for a MAV solution operating in sewer networks.

4.2 *Goals for Phase III*

Payload weight

As discussed in deliverables D26.5 [4] and D26.6 [5] the addition in phase II of the RGBD camera and the Intel NUC increased the payload weight; and our analysis of control failures has shown that the MAV motors often operated near their thrust limits as a result, to maintain the MAV altitude. The control system therefore has little room for manoeuvre when adjusting the relative thrusts as required to control the different axes, and the system struggles to recover from large yaw errors in particular.



Figure 21: Intel RealSense D400 series

While we hope that our numerous flights in the sewers demonstrated that we achieved satisfactory control in this phase, we believe that a reduction in payload weight will significantly improve it further, in addition to increasing the MAV autonomy. Therefore, one of our main goals in phase III will be to reduce the MAV weight, in particular that of the sensor payload. We already know of some options to achieve this, for example using a lighter RGB-D camera such as the [Intel Realsense, D400 series](#) (72g, 10m+ depth range) due for release later this year (

Figure 21).

MAV improvements

Another key goal for phase III will be to develop our MAV prototype into a product adapted to the sewer environment: While some protective measures were implemented in this phase (use of hydrophobe spray, development of motor protections, landing gear, etc.) it is clear that further work is required to select the best materials and designs to operate in the sewers and protect the MAV electronics while reducing the overall platform weight. A reevaluation of the MAV autonomy will also be required. The ARSI consortium will intensify its efforts towards productization in phase III.

Data analysis & GIS geo-referencing

The geo-referencing of inspection data against GIS and maps is a key functionality that will be implemented in phase III and integrated in our post-mission analysis tool [6]. As the ARSI develops into a product, we will work with ARSI partners FCC as well as BCASA teams to implement customer-oriented features such as inspection report generation.